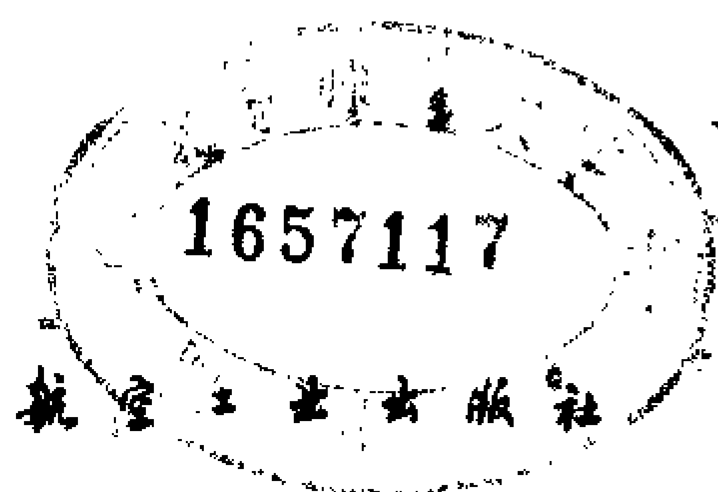
图论及其算法

肖位枢　主编



航空工业出版社

1 9 9 3

（京）新登字161号

内 容 简 介

本书系统地阐述了图的基本理论并结合实际应用介绍其算法。全书分九章：绪论、基本概念、树与割集，平面图、偶图与匹配问题、图的着色、路径问题、回路问题、网络的流。本着理论与实际密切结合的原则，书中除较详细地进行基本理论的分析外，并用较多篇幅作算法的研究和分析，有较多的例题作为算法的应用实例，每章末还附有较多的习题和思考题作为练习，便于加深理解、巩固提高。

本书是一本理论与应用相结合的基础教材，可作为高等工科院校系统工程、管理工程、自动控制、通讯与计算机科学等专业高年级学生或研究生图论课的教材或教学参考书，也可供有关专业的科技人员自学之用。

图论及其算法

肖位枢　主编

航空工业出版社出版发行

（北京市安定门外小关东里14号）

—邮政编码：100029—

全国各地新华书店经售

北京地质印刷厂印刷

1993年7月第1版　　1993年7月第1次印刷

开本：787×1092　1/16　　印张：18.25

印数：1—1800　　字数：453千字

ISBN 7-80046-521-7/TP·036

定价：8.70　元

前　言

图论作为数学的一个分支，已有二百多年的历史，特别是在计算机出现和推动下，图的理论有了迅速的发展，其应用也日愈广泛。现在已成为系统工程、管理工程，计算机科学、通讯与网络理论、自动控制、运筹学以至社会科学等的一种重要的数学工具。图论的应用所以如此广泛，在于它可作为分析处理多种问题的一种较为理想的数学模型，它的算法又可借计算机实现，因而图论与计算机相结合，为图的理论研究和应用开辟了广阔的前景。

本书是编者经过长期的教学实践，在自编教材的基础上，不断加工提炼，反复修改而成。书中除对图的基本理论作了系统地讨论外，并用了较多篇幅介绍图论的各种应用及其算法，举了较多例题作为应用实例及算法说明，目的在于通过对本书的学习，既能掌握图的基本理论，更熟悉一些基本算法，从而可利用计算机解决图论应用的各种实际问题。

鉴于电网络和开关理论内容丰富，已成为图论的一个分支，在工科院校各专业的电工基础或电工学中都有介绍，为了避免重复，不再列入本书内容。因此本书可作为高等工科院校多数专业的通用教材或教学参考书。由于专业要求不同，书中一些理论性较强的内容，如Kuratowsky定理、Vizing定理等的证明可以省略。除理论教学外，如有条件可以结合实际课题安排大作业或上机实习。

本书由肖位枢主编，参加编写的有刘宣（第五章）、肖公信（第四章、第六章）、肖位枢（第一章、第二章、第三章、第七章、第八章、第九章）。

东北大学赵连昌教授在百忙中审阅了本书全稿，并提出了宝贵意见，谨在此表示感谢。

本书在阐述上力求文字简洁、说理清楚、由浅入深、通俗易懂、便于自学，但限于编者水平，书中疏漏或错误之处在所难免，恳请读者多多指正。

肖位枢

1992年12月

目　录

第一章 绪　论

§1.1 图的图形描述

图的严格定义，我们将在第二章中给出，这里想要指出：社会上和自然界很多问题，如果用图形的方式来描述和分析，不仅形象直观，而且会取得很好的效果，从图形分析上升到理论，就是图论。所以，图论是我们研究和分析问题的一种很好的数学工具或数学模型。为了说明这个问题，下面举几个例子。

例 1.1 有三个酒桶A、B、C，如图1.1，容量分别为8、5、3升，现A桶装满了酒，要求只用这三个桶将酒平分为二，问怎样分法。

如果用语言来描述，一种平分的过程或步骤如下：先将A桶的酒倒入C，再将C桶的酒倒入B，然后依次将A倒入C，C倒入B，B倒入A，C倒入B，A倒入C，最后C倒入B，即可得A、B两桶装的酒各为4升。



图 1.1

显然，像这样用语言来描述，前后状态和过程不好记，思路容易混淆，如果用图形来描述就会清晰得多。例如：我们用小圆圈（结点）表示各个状态，在小圆圈旁标上（a、b、c）表示这一状态下A、B、C三个桶内分别装有酒a、b、c升，用0⟶0表示前一状态经过某一过程（步骤）达到下一状态，并在旁边标上$X \to Y$表示这一过程是将X桶的酒倒入Y桶。

起始状态v_0，标记为（8,0,0），表示A桶装满8升酒，B、C均为空桶，下一步骤只能有两种选择，一种是将A倒入C，即A→C，达到状态v_1（5,0,3），另一种是将A倒入B，即A→B，达到状态u_1（3,5,0），如图1.2所示。假设采用第一种选择，状态v_1的下一步可以是C→B，则达到状态v_2(5,3,0)，如果是A→B，则达到状态q(0,5,3)，但是，状态q的下一步只能回到状态v_1或者到状态u_1，这样的过程是不合理的，这就说明状态v_1的下一步只能是经C→B而达到状态v_2，同理v_2的下一步只能是经A→C而达到状态v_3，

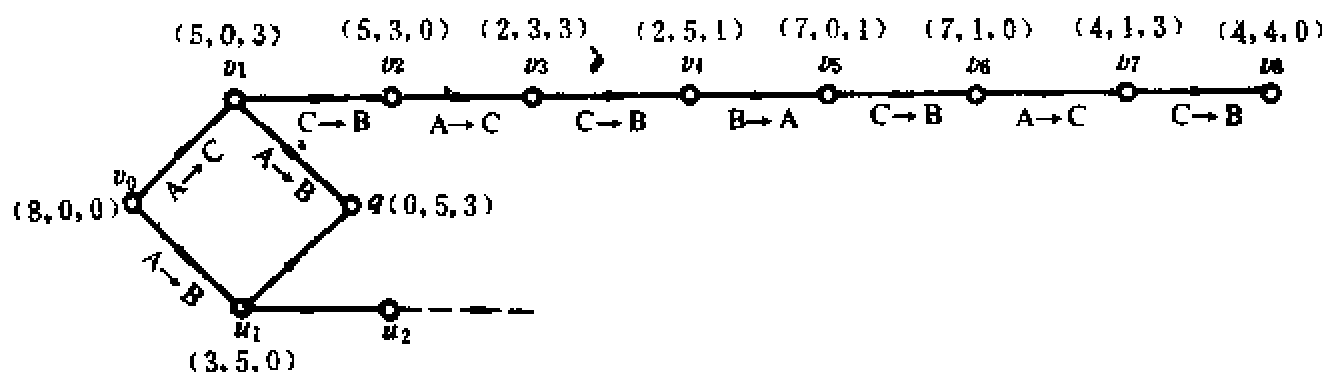


图 1.2

依此类推最后达到状态v_8 (4,4,0)，过程即告结束，这时A、B桶内的酒各为4升。

如果采用第二种选择，即由初始状态经A→B达到状态u_1 (3,5,0)，最后也可实现将酒平分为二，而且所需步骤少于第一种选择，它的图形描述，留给读者。

由此可见，这一问题用图形来描述，不仅形象直观，而且每一步都非常严格，符合逻辑思维和分析的规律，问题怎样解，有多少种解法，哪一种解法最佳，都清楚地呈现出来，使我们对这一问题的解，获得一清晰完整的概念。

例 1.2 试证：任意六个人在一起，其中一定有三个人彼此互相认识，或者有三个人彼此都不认识。

可以用图形描述法证明这个命题。用6个小圆点a，b，c，d，e，f表示任意六个人，如果某两人彼此认识，则在相应的两点之间联一实线，如果两人彼此不认识，则在相应两点之间画一条虚线，因此，任意两点之间如果不存在实线，就一定存在虚线，反之亦然。于是，这一命题的图形表示就是：一定存在一个实线三角形，或者一定存在一个虚线三角形。

六人中的任意一个人，比如a，他对其余五个人来说，最少认识其中三个，否则最少有三个不认识，这两种情况一定有一种而且只能有一种存在。

设a认识其余的三个人，比如b、c、d，则a点与这三个点之间，可联一条实线如图1.3（a）所示。现在来考察b、c、d，如果他们三个人中至少有两人彼此认识，比如b与c，则b与c之间存在一条实线，于是a、b、c三点构成一实线三角形。若b、c、d三人彼此都不认识，则b、c、d构成一虚线三角形，因此，b、c、d之间的任何一种情况都将导致出现实线三角形或者虚线三角形。

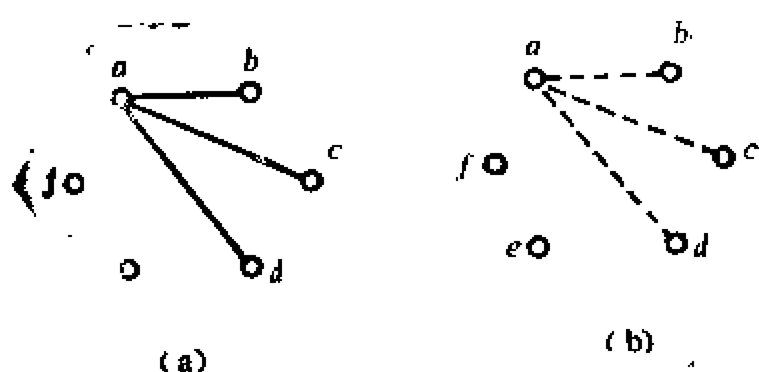


图 1.3

反之，若a对b、c、d都不认识，则a与这三点之间可联一条虚线，如图1.3(b)所示，可用同样方法证明必然存在实线三角形或者虚线三角形。因此命题得证。

例 1.3 图1.4表示某一城市部分街道，图中标的数字表示街道的长度。问从学校A到医院H，怎样走路程最短。

这是一个求最短路径的问题，答案是沿着路径：A（学校）→D→F→G→J→H(医院)，则所走的路程最短。这一答案是怎样得

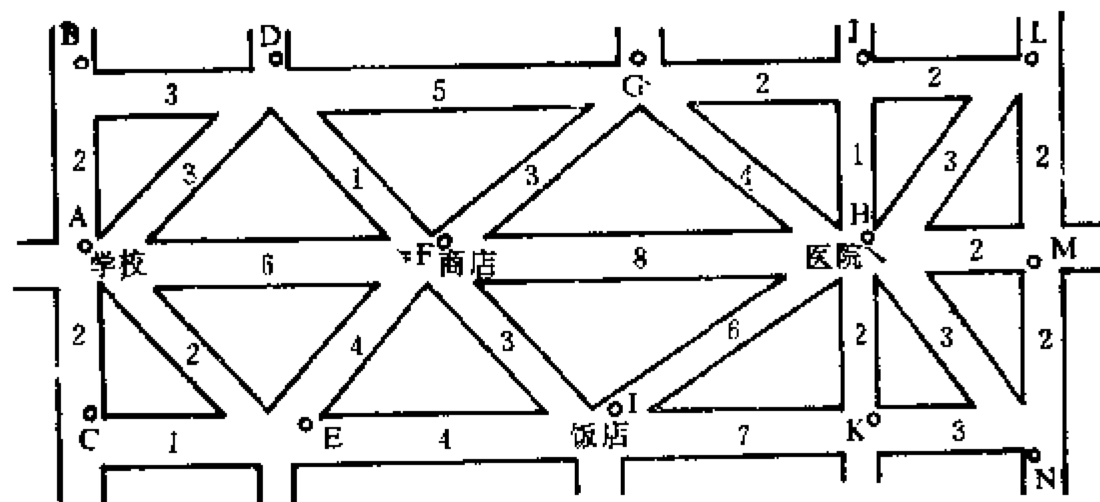


图 1.4

到的，我们将在第七章中详细分析并给出有效算法。

上面列举的例子，只是从几个具体问题说明如何用图形描述和分析，当然不可能勾勒出图论研究问题的全貌，但从中我们可以看到，这些问题都可以看作某些事物以及它们之间存在着的某种联系，图论就是研究事物及它们之间联系的一门学科，正如用函数的图象描述函数一样，在这里，我们用图形来描述所研究的事物及它们之间的联系，即用图形来描述图。由于用图形描述图，形象直观，便于理解，因而是图的一种较好的表达形式，所以图论的很多名词和术语，都采用了图形的名词和术语，例如把事物称之为点，而用边表示事物之间的联系，等等。

图论研究的事物可以是具体的客观实体，也可以是抽象的概念，如同集合中的元素一样，事物之间的联系，可以是静态的关系，也可以是动态的关系，所以，图论中的点和边，不一定需要用平面上的几何点和线来表示，就是说不一定需要用图形来描述图，正如不一定需要用图象描述函数一样。对于图的定义可以进一步抽象化，对它进行高度概括，使它更具有普遍意义，这就是我们以后要加以阐述的。但是，为了对问题的分析更为直观清晰，我们以后在研究图时，仍经常伴以图形描述。

由于图是研究事物及它们之间关系的学科，因此任何一个能用二元关系描述的系统，都可以用图提供数学模型，因此图论模型具有广泛的适用性，特别是计算机出现以后，图论的很多复杂问题，借助计算机得到了圆满的解决，使图论的理论研究和应用，获得了更为迅速的发展。

§1.2 图的拓扑变换

图 1.5 是某地区铁路线路图。图 1.6 是对这个线路图进行简化，得出的一个示意图。

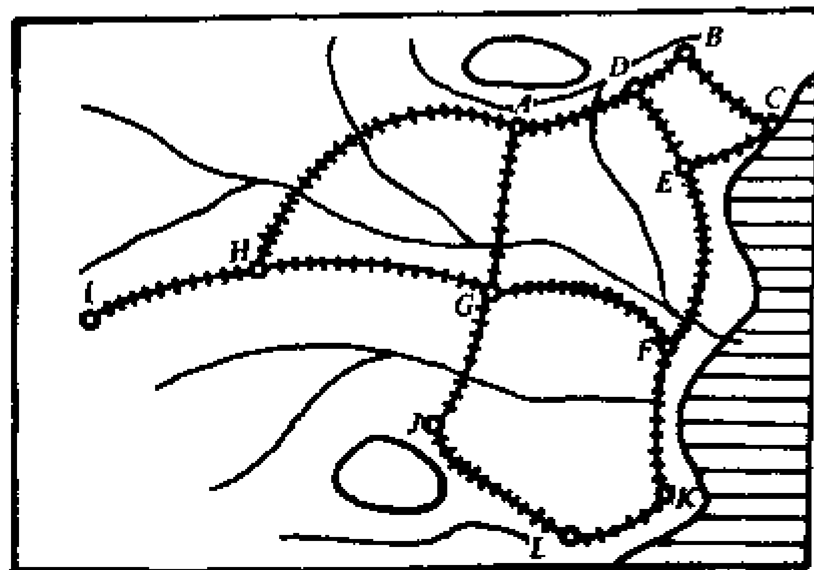


图 1.5

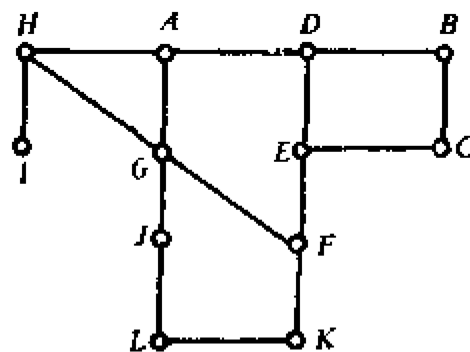


图 1.6

可以看出，示意图与原来的线路图，在外观上有了很大的变动，原来铁路线的长度、方向以及各站的位置，在示意图中都变更了，唯一保持不变的，就是各站之间的联接关系，这一关系表示各站之间的连通性，就是站与站之间，是否有线路连通，并且由某一站至另一站，是否须经别的站。

通常，旅客乘火车时所关心的事情，并不是两站之间的距离和方向等，而是关心在哪一站上车，经过哪些站便可抵达目的地。如果是这样，那么，用示意图代替原来的线路

图，便可达到这一目的，使图形更为清晰简单。

由图 1.5 的实际线路图变成图1.6的示意图，是经过一种几何变换的，这种几何变换的主要特点，就是保持物体原来的连通性，就是说，物体本来是连着的部分，不会因变换而断开，本来不是连着的部分，也不会因变换而接合起来。一个具有保持连通性的变换，称为拓扑变换。两个图形，能够通过拓扑变换变成同一形状的图形，称为是拓扑等价的。

图1.7（a）中的图形是拓扑等价的，同理（b）中的图形也是拓扑等价的，而（c）

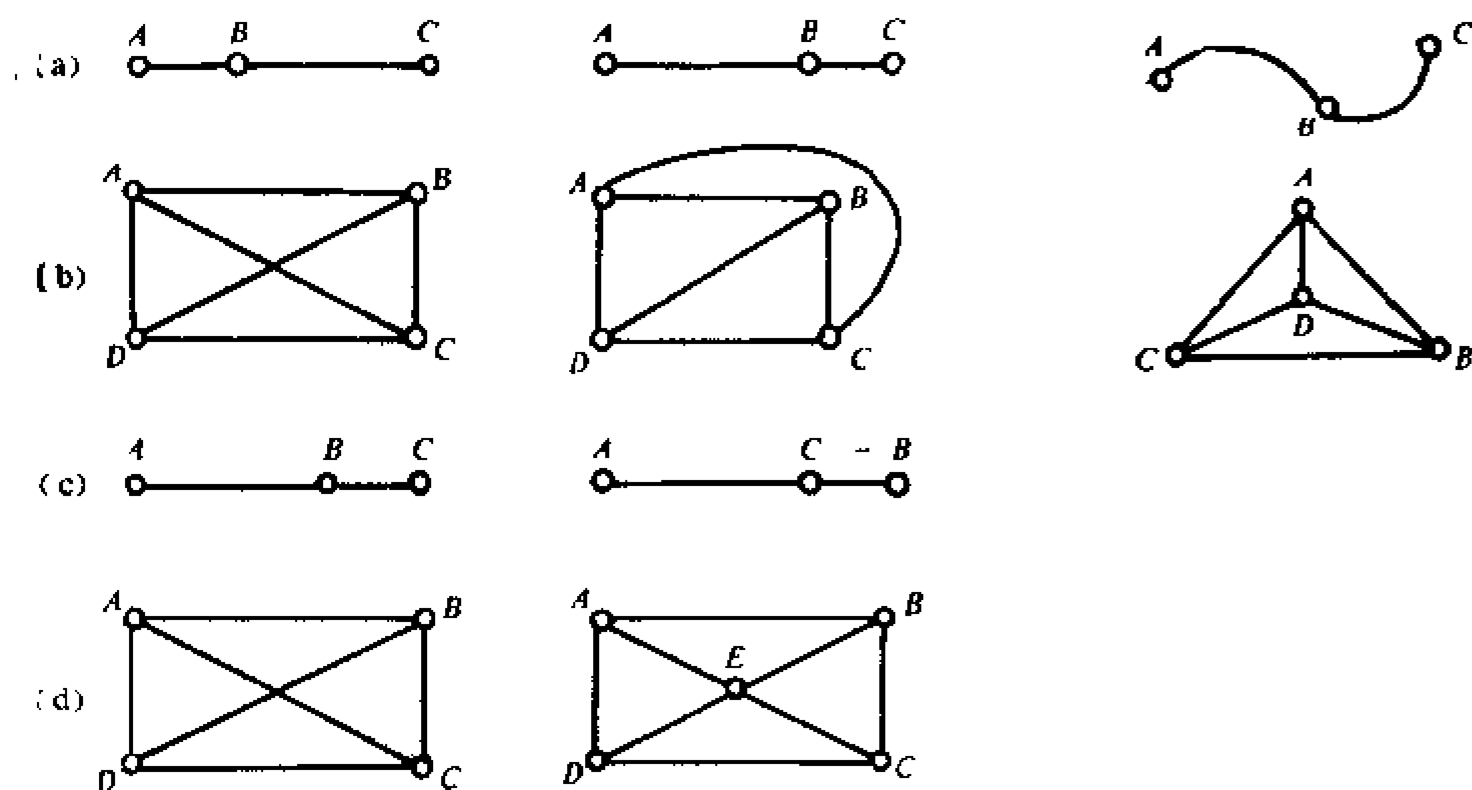


图 1.7

（d）中的图形则不是拓扑等价的。

图的拓扑变换，可以设想为图是由橡皮筋构成的，我们可以任意把它拉长、扭曲，但是却不能把它扯断或接合起来。

用以表示图的图形，也具有这种拓扑变换的特性，所以点和边的位置、形状是无关紧要的，我们关心的只是点与点之间的联接关系，因此，同样性质的图可以画成多种不同的形式，只须保持点和边的关系不变即可。

§1.3 图的计算复杂性

图作为一种数学模型，必然涉及对它的计算，同一个问题，可能有不同的算法，算法不同，效率也会不一样，因此，对一个算法，人们常用计算复杂性去衡量它的效率或计算的难度。评价一个算法计算复杂性的标准，是这个算法需要耗费的时间和空间。如果求解同一问题，算法 A_1比算法 A_2需要的时间和空间少，就说算法 A_1比 A_2好，或 A_1比 A_2的效率高。不言而喻，当求解一个问题时，我们总是希望找到一个效率高的算法。

通常，对一个算法，人们更关心的是时间的耗费，即计算的时间复杂性，而对空间复杂性较少讨论。一个算法的时间复杂性可以简单地表为从输入数据到计算出结果所需的时间或计算的步数（即需要计算少多步才可得出结果）。它是输入数据（初始数据）量的函数。输入数据量常用一个正整数来表示，也称为问题的规模（或大小）。例如，在图论中，

问题规模决定于图的边数或结点数。在计算矩阵乘法时，问题的规模可以用矩阵的阶数来定义。

设输入数据量为 n，则算法A的时间复杂性用$T_A(n)$表示，在不致引起混淆时，可将下标A去掉。当输入量 n 逐渐增大时，时间复杂性的极限称为算法的渐近时间复杂性，它决定了能够处理的问题的最大规模。

在复杂性理论中，函数的量级具有很重要的意义，下面我们给出它的定义。

定义 1.1 给定两个自然数 n 的函数 $F(n)$ 与 $G(n)$，当且仅当存在一个正常数K和一个 n_0，使得 $n \geq n_0$时都有 $F(n) \leq KG(n)$，则称函数 $F(n)$ 以函数 $G(n)$为界，记作 $F(n)=O(G(n))$，或称 $F(n)$是 $O(G(n))$。

这里"O"表示数量级的概念。例如对某个常数$K>0$，一个算法能在 Kn^2 的时间内处理完规模为 n 的输入，就说这个算法的时间复杂性是 $O(n^2)$，读作"n平方级"的。

例 1.4 对于下面三个简单的程序段：

```
(a)  X:=X+1
(b)  FOR i:=1 TO n DO
         X:=X+1;
(c)  FOR i:=1 TO n DO
     FOR j:=1 TO n DO
         X:=X+1;
```

程序（a）只运算一步，执行时间是个常量；程序（b）的语句要执行 n 次，计算的时间与 n 成正比，而程序（c）的计算时间与 n^2成正比，因此，三个程序段计算时间复杂性分别是 $O(1)$、$O(n)$和 $O(n^2)$。

在研究算法复杂性时，算法的渐近时间复杂性是判定算法好坏的一个重要标准，这时，在确定复杂性的量级时，函数的低次项可以忽略，于是多项式（$3n^3+6n^2+n+6$)是 $O(n^3)$ 而$\frac{1}{2}n^2$是 $O(n^2)$，显然，这样表示是很简便的。

当需要比较两个复杂性函数的量级时，常采用下面定义的方法：

定义 1.2 设 $F(n)$与 $G(n)$为自然数 n 的两个函数，令 $\lim\limits_{n\to\infty} F(n)/G(n)=L$，如果

（i）$L=a$，a 为有限正常量，则称 $F(n)$与 $G(n)$同量级。

（ii）$L=0$，则称 $F(n)$的量级比 $G(n)$的量级低。

（iii）$L\to\infty$，则称 $G(n)$的量级比 $F(n)$的量级低。

例 1.5 对以下四种情况，比较两个函数的量级。

（a）设$F(n)=3n^2-4n+2$, $G(n)=\frac{1}{2}n^2$.

则 $L=6$，因此两个函数同量级。

（b）设 $F(n)=\log_2 n$，$G(n)=n$，

则 $L=\lim\limits_{n\to\infty}\frac{\ln n}{n}\cdot\log_2 e=\lim\limits_{n\to\infty}\frac{\log_2 e}{n}=0$

这里我们用了如下法则，即如果

$$\lim_{n\to\infty} F(n)=\lim_{n\to\infty} G(n)=\infty$$

假定有导数 $F'(n)$ 及 $G'(n)$，且极限存在，则

$$\lim_{n\to\infty} F(n)/G(n) = \lim_{n\to\infty} F'(n)/G'(n)$$

因$L=0$，我们即得 $\log_2 n$ 的量级低于 n。

（c）设 $F(n)=a^n$，$G(n)=n^k$，这里 a、k 为任意给定的常量，且都大于1。

令　$U(n)=F(n)/G(n)$，则

$$U(n+1)/U(n)=a(n/(n+1))^k$$

当固定 k 时，我们总可以找到 n 的一个足够大的值，比如说 n_0，使$n>n_0$时，$(n/(n+1))^k \simeq 1$，即

$$U(n+1)\simeq a\cdot U(n)$$

于是当 $n\geqslant n_0$ 时

$$U(n)=a^n/n^k$$
$$\simeq a^n/n_0^k = a^{n-n_0}\cdot U(n_0)$$

则　$L=\lim_{n\to\infty} U(n)=\infty$

由此即得：阶为 n 的指数函数的量级将高于 n 的任何多项式的量级。所以算法为多项式阶的算法是我们所希望得到的，而指数阶的算法，则应尽量避免。

（d）如果 $F(n)$和$G(n)$如（c）所设，且$H(n)=n!$，用类似（c）的同样近似法，读者可以验证 $H(n)$的量级比 $F(n)$和$G(n)$的量级都高。即 n 的阶乘，它的量级高于 n 的多项式，也高于 n 次指数函数的量级。

表1.1列出了六种算法的时间复杂性函数在不同问题规模 n 时，需要的计算步数。

表 1.1

问题规模 n / 时间复杂性	2	8	128	1024
n	2	2^3	2^7	2^{10}
$n\log_2 n$	2	3×2^3	7×2^7	10×2^{10}
n^2	2^2	2^6	2^{14}	2^{20}
n^3	2^3	2^9	2^{21}	2^{30}
2^n	2^2	2^8	2^{128}	2^{1024}
$n!$	2	5×2^{13}	5×2^{714}	7×2^{8766}

计算时间是与计算步数成比例的，设计算机的计算速度为 2^{20}步/秒，时间换算如下：

2^{20}步/秒$\simeq 0.9\times 2^{26}$ 步/分

$\simeq 0.9\times 2^{32}$步/小时

$\simeq 1.3\times 2^{36}$步/日

$\simeq 0.9\times 2^{45}$步/年

$\simeq 0.7\times 2^{52}$步/世纪

设问题规模 $n=128$，则时间复杂性函数为n^2的算法，需要的计算时间只有 1/64 秒，n^3需要的计算时间也不过 2 秒，而 2^n需要的计算时间超过 2^{76}世纪，$n!$需要的计算时间超过 5×2^{662}世纪，显然，后两种算法是不可能在实际有限的时间内完成的。

由此可见，一个算法的时间复杂性函数的量级具有多么重要的意义，它是反映算法性能的重要指标。对于某一个问题而言，如果算法的时间复杂性函数的量级越低，说明算法的效率越高。

可能认为，现代计算机发展突飞猛进，计算速度成百成千倍增加，算法效率的高低已没有多大意义，其实不然，表1.2给出了提高计算机运算速度对列出的算法带来处理能力增加的情况。

表 1.2

算法	时间复杂性	提高速度前单位时间能处理数据量	提高速度 2^3 倍后单位时间能处理数据量	提高速度 2^7 倍后单位时间能处理数据量	提高速度 2^{10} 倍后单位时间能处理数据量
A_1	n	N_1	$8N_1$	$128N_1$	$1024N_1$
A_2	n^2	N_2	$2.8N_2$	$11.3N_2$	$32N_2$
A_3	2^n	N_3	N_3+3	N_3+7	N_3+10
A_4	8^n	N_4	N_4+1	$N_4+2.3$	$N_4+3.3$

从表1.2可以看出，算法 A_1 在同一时间里能处理的输入量增加的倍数与计算机速度提高的倍数相同，算法A_2的处理能力，也比原来增加若干倍，而算法 A_3 和 A_4 就很差，处理能力仅增加若干个，例如算法 A_4，当计算机运算速度提高 1024 倍时，它能处理的数据量，比原先只多不到 4 个。这就说明，对于多项式函数的算法，提高计算机运算速度，对提高计算效率，还能收到较为明显的效果，而对于指数函数的算法，就没有多大意义。因此一般认为，如果一个算法的时间复杂性是以多项式为界的，则是一个有效的算法，对于一个有效的算法．那怕输入的数据量很大，也认为计算机可以处理得了，而对于指数函数的算法，则认为是一个低效率的算法，只要问题的输入量稍微大一些，计算机就无法处理了。

以上是从计算复杂性量级的高低衡量算法的优劣，但是也要指出，当问题的输入量较小时，或者带有不同大小的常量因子时，计算复杂性量级高的算法，可能反而比计算复杂性量级低的算法要好，试看下面的例子。

例 1.6 设算法 A_1、A_2、A_3、A_4、A_5的时间复杂性为：

$T_1(n)=1000\,n$

$T_2(n)=100\,n\log_2 n$

$T_3(n)=10\,n^2$

$T_4(n)=n^3$

$T_5(n)=2^n$

则当：$2\leqslant n\leqslant 9$ 时，A_5比 A_1、A_2、A_3、A_4都好。

$10\leqslant n\leqslant 58$ 时，A_3比 A_1、A_2、A_4、A_5都好。

$59\leqslant n\leqslant 1024$ 时，A_2比其它算法好。

$n>1024$ 时，算法 A_1最好。

所以，研究算法复杂性时，不仅要看复杂性函数的量级，也应注意所含的常数因子以及实际问题的规模。

习题与思考题

1．某人挑一担青菜、牵一条狗、一只羊赶路。途经一条小河，船小一次只容人带狗、

羊、菜三者之一过河，当人不在场时，应避免狗和羊或羊和菜在一起，求过河的方案。

2．选手A、B轮流从10根火柴中抽取火柴，每次只准取 1 或 2 根，不准不取也不准多取，最后取完火柴者获胜。问怎样取方可取胜。

3．有 4 件产品，其中有一件不合格，但它的外形并无差异，只是重量不合标准。问如何以最少的次数用天秤找出这件不合格的产品，并判别出它的重量比标准产品重还是轻。

4．甲、乙两人进行网球比赛，如果某人连胜两局或总计胜三局，他就获胜，比赛即告结束，试分析各种取胜的比赛情况。

5．试判别图 1.8 中的图形，哪些是彼此拓扑等价的。

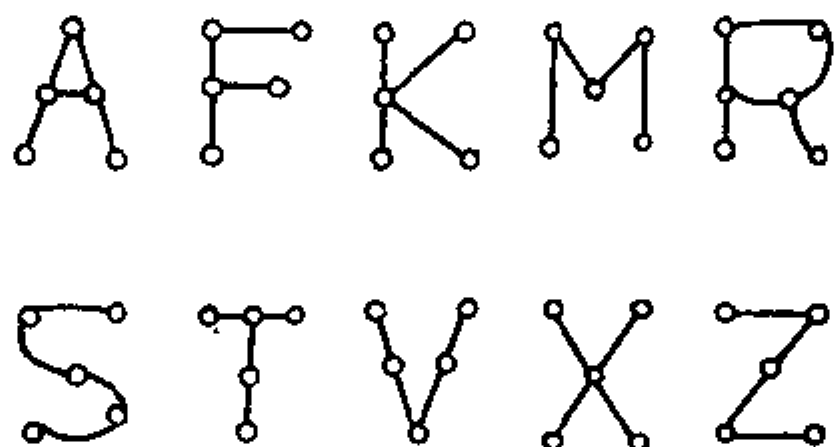


图 1.8

6．图 1.9 中的任意两个图形，是否都是拓扑等价的。

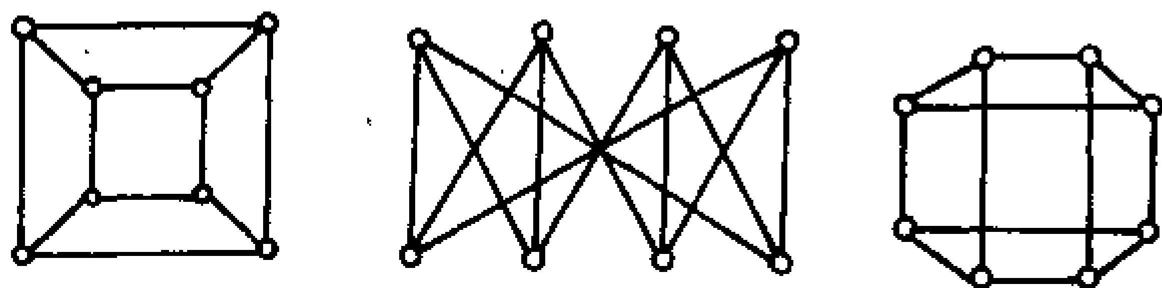


图 1.9

7．如果对于某一 $k>0$ 及存在 $n_0>0$，当 $n>n_0$时，有 $F(n)\geqslant k$ 且存在常数 $C_1>0$ 和 $C_2>0$，使得当 $n>n_0$时，有

$$G(n)\leqslant C_1F(n)+C_2$$

证明$G(n)$是$O(F(n))$。

8．对下面的函数对，确定最小的整数值 $n_0>0$，使得当 $n\geqslant n_0$ 时每对中的第一个函数恒大于等于第二个函数。

（1）n^2，$10n$。

（2）2^n，$2n^3$。

（3）$n^2/\log_2 n$，$n(\log_2 n)^2$。

（4）$n^3/2$，$n^{2.81}$。

9．如果存在一个正的常数 c，使得一切 $n\geqslant 0$，有 $F(n)\leqslant CG(n)$，则记为 $F(n)\leqslant G(n)$。试证明：若 $F_1\leqslant G_1$和 $F_2\leqslant G_2$，必有

$$F_1+F_2\leqslant G_1+G_2$$

关系“$\leqslant$”还有一些什么性质？

第二章　图的基本概念

§2.1　图 与 子 图

一、有向图与无向图

第一章讲了如何用图形描述图，同时也指出图如同函数一样，也是一个数学抽象。这里我们从集合论的角度来给图下定义。

定义 2.1　一个有向图D是一个三元组(V,E,f)，其中V是一个非空集合，它的元素称为有向图D的结点，E是一个集合，它的元素称为有向图D的弧(边)，f是从E到$V\times V$上的一个映射（函数）。

例 2.1　设$V=\{a,b,c,d\}$，$E=\{e_1,e_2,e_3,e_4,e_5,e_6\}$，且$f(e_1)=\langle a,b\rangle$，$f(e_2)=\langle c,b\rangle$，

$$f(e_3)=\langle b,c\rangle,\ f(e_4)=\langle c,d\rangle,$$
$$f(e_5)=\langle d,b\rangle,\ f(e_6)=\langle d,d\rangle。$$

则$D=(V,E,f)$是一个有向图，它的图形如图 2.1 所示。

例 2.2　设有向图$D=(V,E,f)$的图形如图 2.2 所示，则可得：

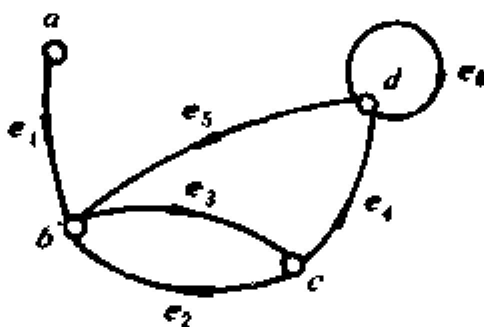


图 2.1

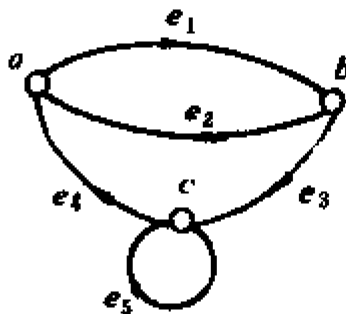


图 2.2

$$V=\{a,\ b,\ c\}$$
$$E=\{e_1,\ e_2,\ e_3,\ e_4,\ e_5\}$$
$$f(e_1)=\langle a,\ b\rangle,\ f(e_2)=\langle b,\ a\rangle$$
$$f(e_3)=\langle b,\ c\rangle,\ f(e_4)=\langle c,\ a\rangle,\ f(e_5)=\langle c,\ c\rangle.$$

从定义可知，E中的元素总是与V中的序偶有着对应的关系，因此可用V中的序偶代替E中的元素。如例2.2，可写成$E=\{\langle a,\ b\rangle,\ \langle b,\ a\rangle,\ \langle b,\ c\rangle,\ \langle c,\ a\rangle,\ \langle c,\ c\rangle\}$。一个有向图$D$，可简记为$(V,E)$。仿此可以对无向图定义如下：

定义 2.2　一个无向图G记作$G=(V,E)$，其中V是一个非空集合，它的元素称为图的结点，E是V中的无序偶集合，它的元素称为图的边。

例 2.3　设$V=\{a,\ b,\ c,\ d\}$，$E=\{(a,\ b),\ (b,\ b),\ (b,\ c),\ (c,\ d),(d,\ a),\ (a,\ d)\}$，则

$G=(V, E)$是一个无向图，它的图形如图 2.3所示。

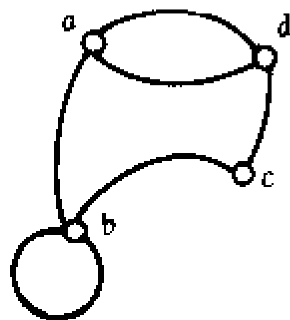


图 2.3

无向图与有向图的区别，在于E中的元素是无序偶还是有序偶。从图形上看，无向图的边是无向边，有向图的边是有向边。一个有向图$D=(V,E)$，如果将E中的有序偶元素改为无序偶元素，变成无向图G，称为有向图D的基础图（或底图）。从图形上看，将有向图边的箭头去掉，得到的无向图就是它的基础图。有向图和无向图，统称为图。

在图中，边e的两个端点a，b称为e的关联点，并称点a和b是邻接的，不与任何结点邻接的结点称为孤点，只有孤点的图称为零图，只有一个孤点的图称为平凡图。结点集合V是有限集合的图称为有限图，否则称为无限图，今后我们只讨论有限图，并简称为图。从结点到自身的边称为自环；在有向图中两结点之间同一方向的边称为平行边，对于无向图，关联于相同两结点之间的边称为平行边；含平行边的图称为多重边图，不含平行边和自环的图称为简单图；结点的数目称为图的阶。在无向图中，与结点v关联的边的数目称为结点v的次数，记作$deg(v)$；在有向图中，从结点v引出的弧的数目称为v的引出次数，记作$deg^+(v)$，引向v的弧的数目称为v的引入次数，记作$deg^-(v)$，v的引出次数与引入次数的和称为结点v的次数，记作$deg(v)$。

例 2.4 如图2.3的无向图，$deg(b)=4$, $deg(a)=3$。如图2.1的有向图,$deg^+(b)=1$, $deg^-(b)=3$，故$deg(b)=deg^+(b)+deg^-(b)=1+3=4$，而$deg^-(d)=2$，$deg^-(d)=2$，故$deg(d)=4$。

一个图常用(n, m)表示，其中n为图的结点数，m为图的边数，即$n=|V|$, $m=|E|$。

无论是有向图还是无向图，下面两条定理，都表明了结点次数的特性。

定理 2.1 任意一个图(n, m)，结点次数的总和等于边数的二倍，即

$$\sum_{i=1}^{n} deg(v_i)=2m \tag{2.1}$$

证：因一条边与两个结点关联，出现一条边就使结点的总次数增加2，因此边数的二倍就是结点的总次数。 ∎

定理 2.2 任意一个图，次数为奇数的结点数必为偶数。

证：设V_1和V_2分别为图中次数为奇数和偶数的结点集合，因而有$V_1 \cup V_2=V$，$V_1 \cap V_2=\phi$。则

$$\sum_{v\in V} deg(v)=\sum_{v\in V_1} deg(v)+\sum_{v\in V_2} deg(v)=2m$$

因$2m$是偶数，$\sum_{v\in V_2} deg(v)$是偶数之和必为偶数，故$\sum_{v\in V_1} deg(v)$也应是偶数，但V_1中每个结点次数都是奇数，因此必须有偶数个奇次结点，才可能使它们的次数和为偶数，命题得证。

二、正则图与完全图

定义 2.3 一个简单无向图$G=(V, E)$.

1）如果每个结点都有相同次数d，称G为d次正则图。

2）如果任意两结点之间都有边连接，称G为完全图。n个结点的无向完全图记作K_n。

图2.4(a)是三次正则图，(b)是5阶完全图。完全图一定也是正则图，n阶完全图是($n-1$)次正则图。



(a)　　　　　　　　(b)

图 2.4

定理 2.3　n阶完全图K_n的边数

$$m=\frac{1}{2}n(n-1) \tag{2.2}$$

证：n级完全图任一结点的次数均为($n-1$)，故结点的次数和

$$\sum_{v\in V}deg(v)=n(n-1)=2m$$

$$m=\frac{1}{2}n(n-1)$$ ∎

用图模拟一个系统时，很多情况下希望将附加信息标在图上，这种附加信息称为权，它可能是数字、符号或某种量。标有权的图称为带权图，也叫网络。权如果标在边上称为边权图，如果标在结点上称为点权图，当然一个带权图既可以是点带权同时又是边带权。

三、子图与补图

定义 2.4　设图$G=(V, E)$，如果有图$G'=(V', E')$，满足：

1）$V'\subseteq V$及$E'\subseteq E$，则称G'是G的子图。如果$V'\subset V$且$E'\subset E$，则称G'是G的真子图。

2）$V'=V$及$E'\subseteq E$，则称G'是G的生成子图。

3）V'是V的非空子集，对任意$v_1, v_2\in V'$，如$(v_1, v_2)\in E$必有$(v_1, v_2)\in E'$，则称G'是G的导出子图。

如图2.5所示，(b)，(c)，(d)都是(a)的子图，(b)，(d)是(a)的真子图，(c)是(a)的生成子图，(d)是(a)的导出子图，但(b)不是(a)的导出子图。

定义 2.5　由图G的所有结点和能使G成为完全图的添加边所构成的图，称为G的补图，记作$\bar{G}$。

图2.6(a)，(b)，(c)给出了补图的三个例子。由定义可知，补图是可逆的，即如果G_1是G_2的补图，则G_2也是G_1的补图。一个图与它的补图有共同的结点但无共同的边。一个图与它的补图叠合在一起，得到一个完全图。因此，完全图的补图是零图。n个结点的图G，它的边数m与它的补图的边数m'有如下关系：

$$m+m'=\frac{1}{2}n(n-1)$$

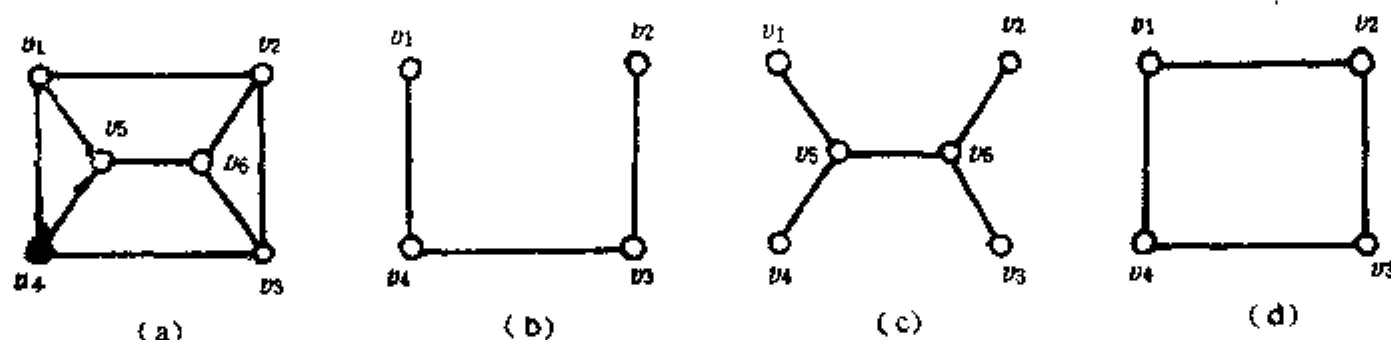


图 2.5

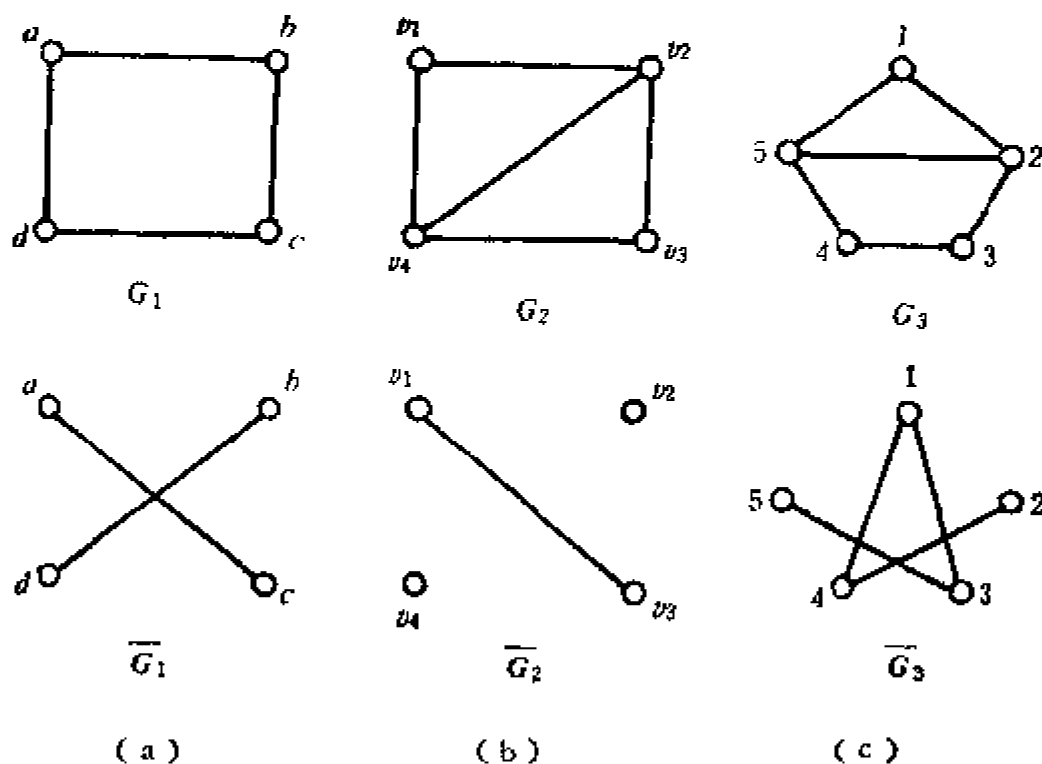


图 2.6

四、图的运算与同构

我们把图定义为两个相关集合的偶对，因此对图也可进行各种集合的运算，上面讲的补图，就是对图进行求补运算，此外还可进行交、并、差等运算。首先我们约定：如果两个图没有公共结点，则称这两个图不相交，如果两个图没有公共边，则称这两个图不重边。下面的定义都假定两个图无孤点。

定义 2.6 设$G_1=(V_1, E_1)$，$G_2=(V_2, E_2)$ 是两个图。

1）G_1与G_2的交，构成一个新图$G_3=(V_3, E_3)$，记作$G_3=G_1\cap G_2$。其中 $V_3=V_1\cap V_2$，$E_3=E_1\cap E_2$。

2）G_1与G_2的并，构成一个新图$G_4=(V_4, E_4)$，记作$G_4=G_1\cup G_2$。其中$V_4=V_1\cup V_2$，$E_4=E_1\cup E_2$。

3）G_1与G_2的差，构成一个新图$G_5=(V_5, E_5)$，记作$G_5=G_1-G_2$。其中 $E_5=E_1-E_2$，$V_5=(V_1-V_2)\cup\{E_5$ 中的边所关联的结点$\}$。

4）G_1与G_2的环和，构成一个新图 $G_6=(V_6, E_6)$，记作 $G_6=G_1\oplus G_2$，且有

$$G_6=(G_1\cup G_2)-(G_1\cap G_2)$$

图 2.7 给出了图 G_1 与 G_2 上述四种运算的图例。

除了图的集合运算之外，图还有与其图形密切联系的一些运算，如图的增添、删除和分割。所谓删除就是去掉图的某些边或结点，如果删除边，与边关联的结点仍然保留，如果删除结点，与结点关联的边要一起删去。删除运算实际上就是求图的子图。所谓增添就是在原来图的某些结点之间添加新的边，相当于结点相同的图的并运算。图的分割是图的

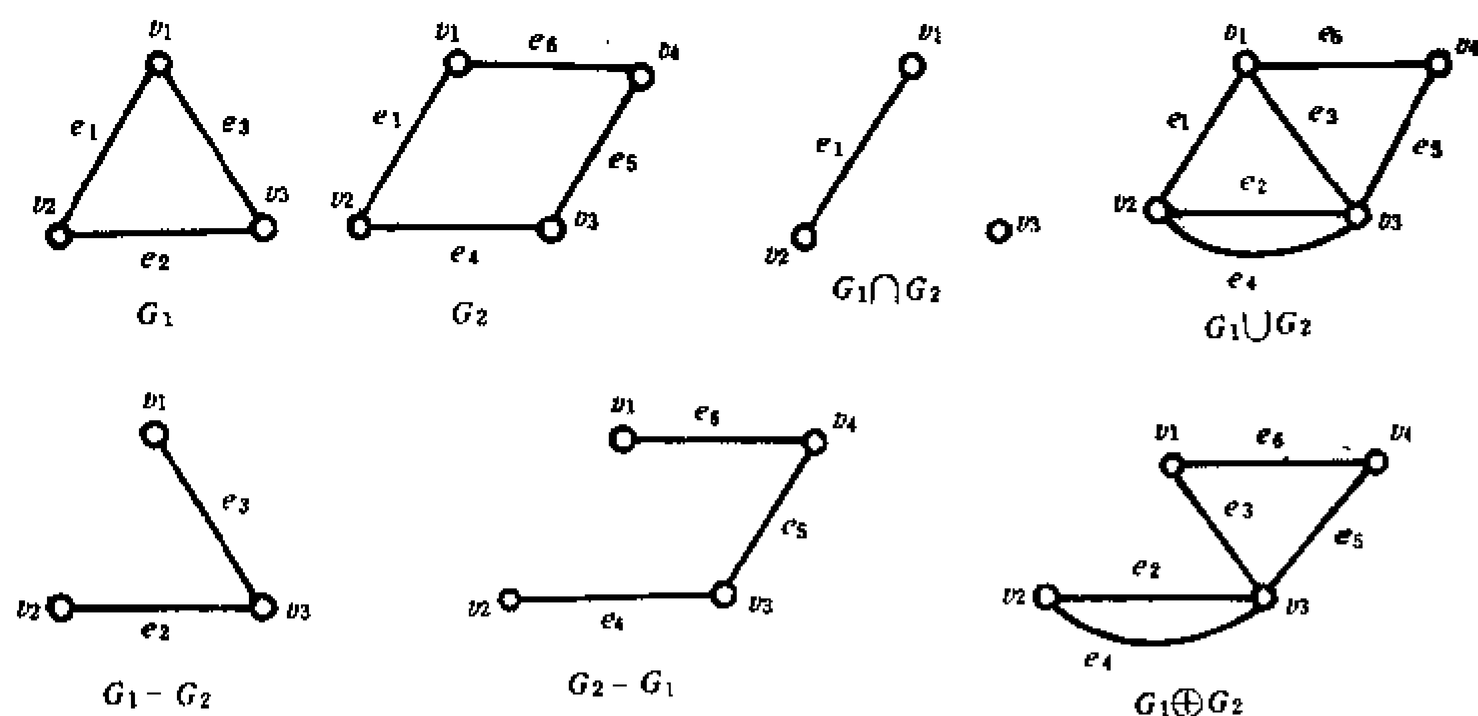


图 2.7

一种很重要的运算，它对分析研究图有着很重要的作用，我们将在第三章中详细讨论。

上一章我们讲了图形的拓扑等价性，因此一个图可以画出不同形状的图形，或者说表面上看完全不同的图形可能表示同一个图。为了判别不同的图形是否反映同一个图的性质，我们给出同构图的定义如下：

定义 2.7 设有两个图 $G=(V, E)$及$G'=(V', E')$，如果结点集合之间存在一一对应关系，而且对应结点之间的边也有一一对应的关系，则称图G和图G'是同构的，记作$G\cong G'$。

对于有向图的同构，对应边的方向要求相同。

图2.8（a)，（b)，（c)表示了三对同构无向图，（d)，（e)为二对同构有向图。例

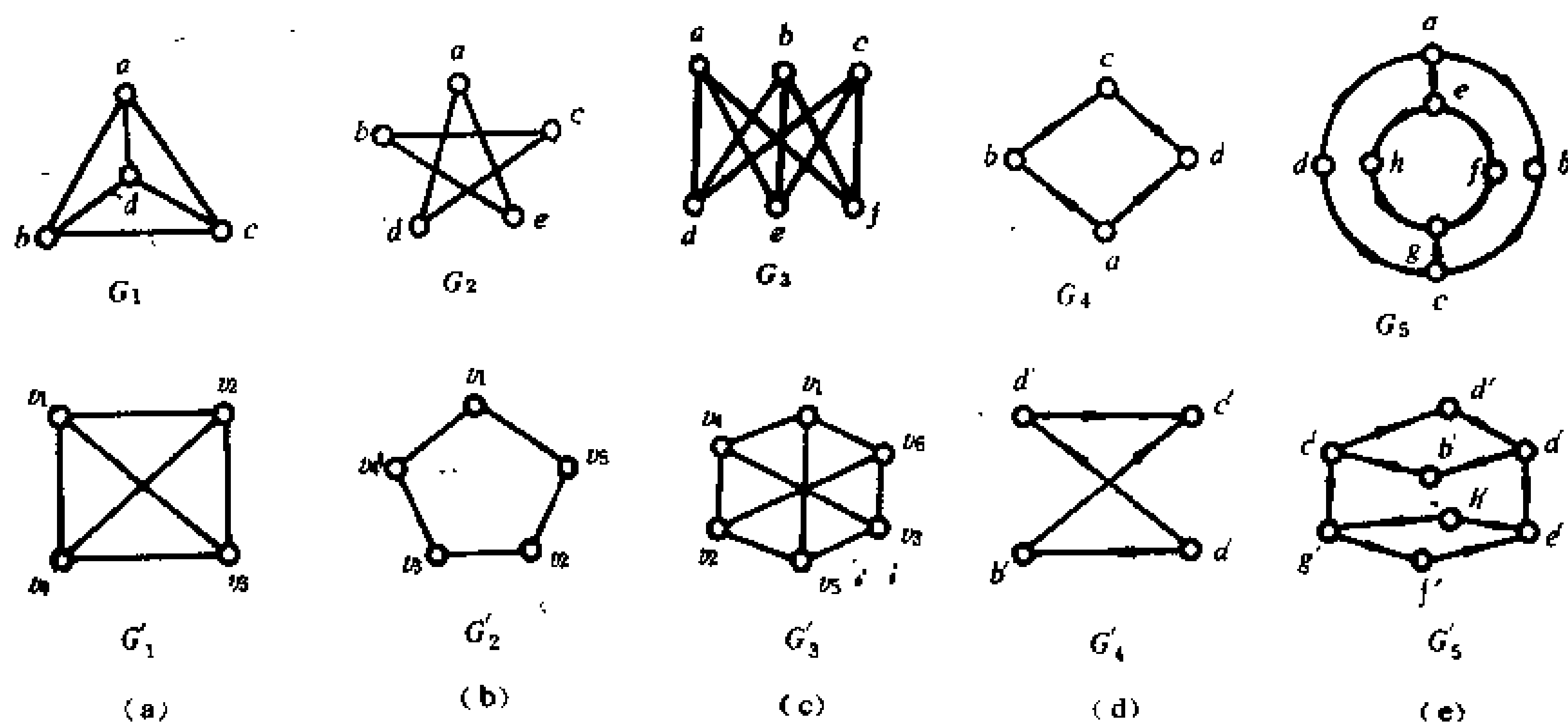


图 2.8

如（d)，两个有向图结点的对应关系如下：

$$a — a',\ b — b',\ c — c',\ d — d'$$

对应结点之间的弧的对应关系如下：

$\langle a,b\rangle-\langle a',b'\rangle$, $\langle b,c\rangle-\langle b',c'\rangle$, $\langle a,d\rangle-\langle a',d'\rangle$, $\langle d,c\rangle-\langle d',c'\rangle$.

因此，除了结点或边标号不同之外，同构图是恒同的，同构关系是一等价关系，同构的图具有完全相同的性质，对一个图的任何命题，也适用于它的同构图。

但是，根据定义判定两个图是否同构是很麻烦的，能否找到一个简单而又充分有效的法则判定两个图是否同构，仍然是图论中尚待解决的问题。从定义可知，两个图同构，必须满足如下条件：(1)两个图的结点数相等，(2)两个图的边数相等，(3)次数相同的结点数相等。这三个条件中任何一条不满足，两个图不会是同构的，因为不可能得到一一对应的关系。但是三个条件都满足的两个图，不一定同构。图 2.9 (a)，(b) 就是一个例子。所以上述三个条件，只是同构的必要条件，而不是充分条件。

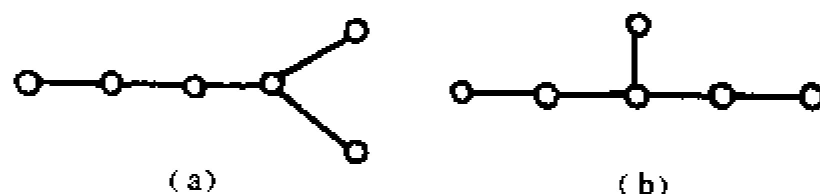


图 2.9

§2.2 图的连通性

一、通路与回路

图的连通性是图论的一个基本概念，这里我们首先讨论有向图，提出的一些定义、定理和分析方法，都可推广到无向图中去。

定义 2.8 在有向图$D=(V,E)$中，把边的一个序列$\langle v_{i_1},v_{i_2}\rangle$，$\langle v_{i_2},v_{i_3}\rangle$ …，$\langle v_{i_{k-1}},v_{i_k}\rangle$称为从结点$v_{i_1}$到$v_{i_k}$的一条路径，并称$v_{i_1}$为路径的起点，$v_{i_k}$为路径的终点。

一条路径也可以用结点的序列来表示，如：

$$v_{i_1},v_{i_2},v_{i_3},\cdots,v_{i_{k-1}},v_{i_k}.$$

由定义可知，一条路径经过的结点和边是允许重复的。如图2.10从结点a到e的路径有：

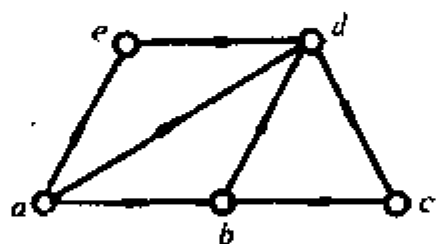


图 2.10

$$p_1=ae$$
$$p_2=ade$$
$$\vdots$$
$$p_i=abdcbde$$
$$p_k=adcbde$$

其中p_i是一条有边重复的路径，p_k是有结点重复的路径。显然在寻找一条路径时总是希望经过的边或结点最好不要重复，于是有如下定义。

定义 2.9 边不重复但结点可以重复的路径称为简单路径。结点不重复的路径称为基本路径。

在基本路径中，结点不重复，边当然不会重复，它是各种路径中最有意义的，一般情况下我们寻求的都是基本路径。

利用通路的概念，我们可以把一些不容易用数学式子描述和解决的问题，用图的方法

来描述和解决，第一章的例 1.1 就是一个例子。

定义 2.10 一条起点和终点相同的路径称为回路，没有重复边的回路称为简单回路，没有重复结点的回路称为基本回路。

一条基本路径是不含回路的，含有回路的路径肯定不是基本路径。因此对任意一条路径，如果删去其中的回路就可得到一条基本路径。同理，在一个回路中，删去它内部所包含的回路，也可得到一条基本回路。

定义 2.11 一条路径（回路）所含边的数目，称为这条路径（回路）的长度。

定理 2.4 在一个有 n 个结点的有向图中，任何基本路径的长度都小于等于$(n-1)$。

证：因为基本路径中的结点是不重复的，如果基本路径通过图的所有 n 个结点，这是最可能长的基本路径，其长度为（$n-1$），如果基本路径没有通过所有 n 个结点，其长度小于（$n-1$），因此不可能有长度大于（$n-1$）的基本路径。 ▮

推论：在 n 阶有向图中，任何基本回路的长度不大于 n。

定义 2.12 在有向图中，如果存在从结点 u 到结点 v 的路径，则称结点 u 可到这 v。

结点之间的可达性，只表明结点之间存在路径，既不考虑有多少条路径，也未考虑路径的长度。但结点之间的可达性与基本路径却有着密切的关系。

定理 2.5 在有向图中，如果从结点 u 可到这 v，则一定存在一条从 u 到 v 的基本路径。

证：因可从 u 到达 v，故从 u 到 v 至少存在一条路径，现选取一条最短路径（即长度最小的路径），下面证明它就是基本路径。

设这条路径用结点序列表示如下

$$u_1(u),\ u_2,\ u_3,\ \cdots,\ u_i,\ u_{i+1},\ \cdots u_j,\ u_{j+1},\ \cdots u_s,\ u_{s+1}(v).$$

用反证法，假设这条路径不是基本路径，则必有结点重复出现，设 $i=j$时，u_i与 u_j是同一结点，即 $u_i=u_j$，把 u_i到 u_j之间的结点序列删除，路径变为：

$$u_1(u),\ u_2,\ u_3,\ \cdots u_i,\ u_{j+1},\ \cdots,\ u_s,\ u_{s+1}(v),$$

仍然是从 u 到 v 的路径，但它比原先路径更短，与假设原先路径是最短路径矛盾，因此原先选择的一条最短路径不会有重复的结点，它是一条基本路径。 ▮

定义 2.13 从结点 u 到 v 最短路径的长度，称为 u 到 v 的距离，记作 $d(u,v)$。如果从 u 到 v 不存在任何一条路径，则称 u 不可到这 v，并称 u 到 v 的距离无定义或无限大。

二、图的连通类型

定义 2.14 在有向图中，如果改变某些弧的方向，能从结点 u 到达 v（或从 v 到这 u），则称结点 u 与 v 是连接的，并称结点 u 与 v 之间存在一条半路径。

在图2.11中，不存在 u 到 v（或 v 到 u）的路径，但 u 与 v 之间存在一条半路径，因为如果改变边 e_3 的方向，则可从 u 到达 v。

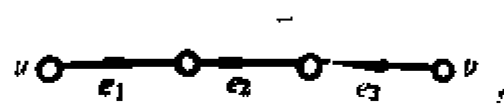


图 2.11

定义 2.15 在有向图D中，对任意两个结点 v_i 与 v_j，

1）如果 v_i与 v_j可以互相到这，则称D为强连通图。

2）如果 v_i不可到这 v_j，就一定可从 v_j 到这 v_i，则称D是单向连通图。

3）v_i 与 v_j是连接的，则称D是弱连通图。

4）不满足以上条件的图称为不连通图。

图 2.12 中，（a）是强连通图，（b）是单向连通图，（c）是弱连通图，（d）则是不连通图。

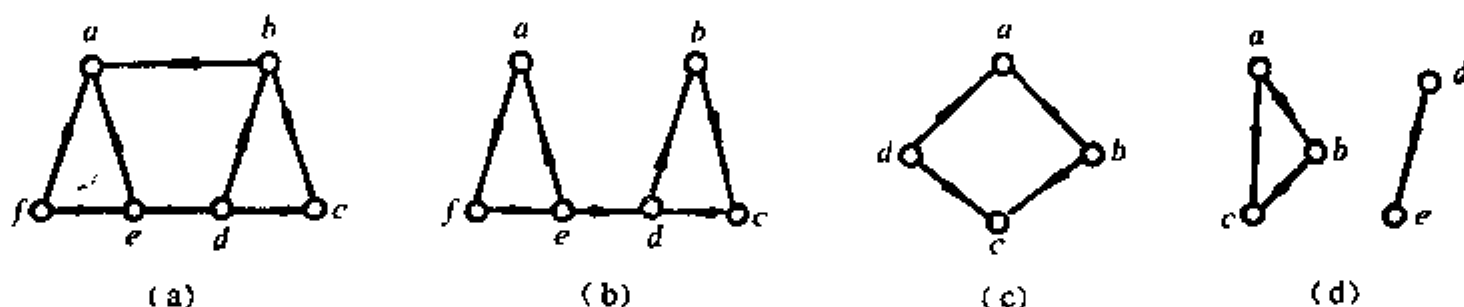


图 2.12

连通图的类型是根据连通性的强弱划分的，强连通图的连通性最强，显然强连通必然也是单向连通，单向连通必然也是弱连通，但其逆不成立。

判别一个有向图是哪一类连通图，如果采用检查任意两点间路径的方法，当结点数 n 较大时，是很繁琐的，下面介绍一个有效的判别定理，先给出一个定义。

定义 2.16 在有向图D中

1）通过所有结点的回路，称为D的完备回路。

2）通过所有结点的路径，称为D的完备路径。

3）通过所有结点的半路径，称为D的完备半路径。

定理 2.6 一个有向图D是强连通的，当且仅当它有一条完备回路。

证：充分性：设D有一完备回路。

$$C = v_1, v_2, \cdots, v_i, v_{i+1}, \cdots, v_j, v_{j+1}, \cdots, v_t, v_1。(l \geqslant n)$$

对D的任意两个结点 v_i, v_j，必然在回路 C 上，则 $v_i, v_{i+1}, \cdots, v_j$ 是从 v_i 到 v_j的一条路径，即 v_i 可到达 v_j。而 $v_j, v_{j+1}, \cdots, v_t, v_1, v_2, \cdots v_i$ 则是 v_j到 v_i的一条路径，故 v_j亦可到达 v_i。因 v_i、v_j 是D的任意两个结点，即D的任意两点都可互相到达，所以D是强连通的。

必要性：设D是强连通的，把D的全部结点列出如下：

$$v_1, v_2, \cdots, v_{n-1}, v_n$$

则必然存在 v_1到 v_2的路径 p_1，v_2到 v_3的路径 p_2，…，v_{n-1}到 v_n的路径 p_{n-1}，及 v_n 到 v_1的路径p_n。把这些路径按次序排列如下

$$p_1, p_2, \cdots, p_{n-1}, p_n$$

显然达是一条从 v_1到 v_n 又回到 v_1 并且通过D的所有结点的完备回路。 ▮

推论 1：一个有向图D是单向连通的，当且仅当它有一条完备路径。

推论 2：一个有向图是弱连通的，当且仅当它有一条完备半路径。

例 2.5 图2.12中，（a）存在一条完备回路：

$$afedbcdba$$

所以（a）是强连通图。（b）存在一条完备路径：

$$afedbc$$

所以（b）是单向连通图。同理可判定（c）是弱连通图。

定义 2.17 在有向图D中，最大的强连通子图 D'称为D的强连通分图，简称强分图。

所谓最大的强连通子图D'，是指 D'是强连通的子图，且D不再有包含 D'的强连通子

图。

如图2.13，下列子图都是强分图：

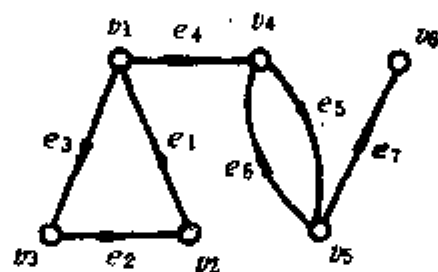


图 2.13

$$D_1=(\{v_1, v_2, v_3\}, \{e_1, e_2, e_3\})$$
$$D_2=(\{v_4, v_5\}, \{e_5, e_6\})$$
$$D_3=(\{v_6\}, \phi)$$

为了表示方便，强分图常用它的结点集合表示，如上面三个强分图表示为$\{v_1, v_2, v_3\}$，$\{v_4, v_5\}$，$\{v_6\}$。

如果以两个结点是否处在一个强分图中作为这两个结点的一种关系，那么这是一个等价关系。因为结点自身必然在同一个强分图中，因此关系是自反的，显然也是对称的。如果v_i与v_j在同一个强分图中，v_j与v_k在同一个强分图中，则v_i与v_k必也在同一个强分图中，因为v_i, v_j, v_k都是互相可到达的，所以具有传递性。因此一个强分图就是一个等价类，强分图的集合就是等价类的集合。它产生有向图D结点集合的一个划分，由划分的性质，立刻得出如下定理。

定理 2.7 有向图D的每一个结点必在一个且仅在一个强分图中。

推论：有向图的各强分图彼此是不相交的。

由于强分图内的结点是彼此可达的，它们与其它强分图中结点的关系，可以用一个结点来代表，这样就可将一个有向图化简为一个更为简单的有向图，这就是将图压缩的概念，定义如下：

定义 2.18 设有向图$D=(V, E)$有强分图$D_1, D_2, \cdots, D_k$，按下面条件构造一个新的有向图$D^*=(V^*, E^*)$：

1）$V^*=\{D_1, D_2, \cdots, D_k\}$

2）E^*的构造如下：当且仅当$i\neq j$时，存在结点$u\in D_i$及$v\in D_j$且在D中有$\langle u, v\rangle\in E$，则从D_i到D_j引一条弧$\langle D_i, D_j\rangle$。

则称有向图D^*是有向图D的压缩。

例 2.6 求图 2.14 的有向图D的压缩D^*。

解：可求出D的强分图如下：

$$D_1=\{a, b, c\}, D_2=\{d\}, D_3=\{e\},$$
$$D_4=\{f, g, h\}, D_5=\{i, j, k\}, D_6=\{l, m\}$$

每个强分图作为压缩图的结点，得到压缩图D^*的结点集合

$$V^*=\{D_1, D_2, D_3, D_4, D_5, D_6\}$$

在D_1与D_3中，有$c\in D_1$及$e\in D_3$且$\langle c, e\rangle\in E$，故$\langle D_1, D_3\rangle\in E^*$。同理可得$\langle D_1, D_4\rangle$，$\langle D_2, D_3\rangle$，

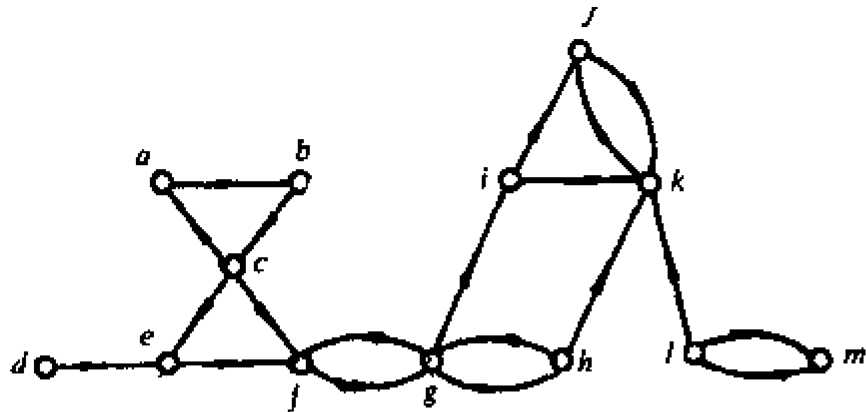


图 2.14

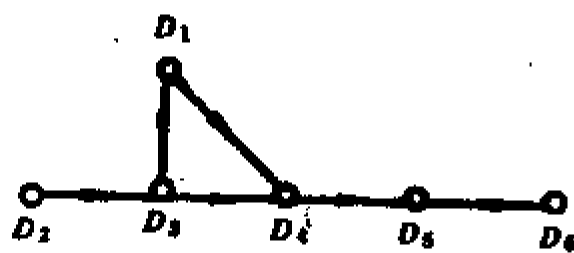


图 2.15

$D_4\rangle$, $D_5\rangle$, $\langle D_3, D_4\rangle$, $\langle D_6, D_5\rangle \in E^*$，于是

$E^*=\{\langle D_1, D_3\rangle, \langle D_1, D_4\rangle, \langle D_2, D_3\rangle, \langle D_4, D_5\rangle, \langle D_3, D_4\rangle, \langle D_6, D_5\rangle\}$压缩图 D^*的图形如图2.15。

三、无向图的连通性

对于无向图，同样可定义路径、基本路径，回路、基本回路、可达性等概念，这里不赘述。但要指出，在无向图中，如果存在结点 u 到 v 的路径，也必然存在 v 到 u 的路径，因此两点之间的可达性是对称的，一般用连通性代替，由两点之间的连通性可推广到图的连通性，得到如下定义。

定义 2.19 无向图$G=(V,E)$，如果任意两点之间至少有一条路径，则称G是连通图。否则称为非连通图或分离图。

因此无向图的连通性，只有连通与不连通两种情况。对于非连通图，可以看作是由若干个连通分支（子图）组成。如图 2.16的无向图含有两个连通分支。

定理 2.8 有n个结点的连通无向图G，至少有 $n-1$条边。

证：用归纳法证明如下：

当结点数$n=1$或2时，命题显然成立。设结点数为（$n-1$）时命题成立，即图至少有（$(n-1)-1$）$=n-2$条边，现增加一个结点v，由于图仍然是连通的，所以v必须与原来（$n-1$）个结点中的至少一个结点连接，即至少要增加一条与v关联的边，故n个结点至少有（$n-2)+1=n-1$ 条边。 ▮

在无向图中，有一些特殊的点和边，称为割点和割边，它们对图的结构有较大的影响，定义如下：

定义 2.20 在连通无向图G中：

1）如果去掉一个结点v及与v关连的边，图将不连通，则称结点v为图的割点（关节点)。

2）如果去掉一条边图成为不连通，则称这条边为图的割边（桥)。

图 2.17，v_3，v_4是割点而e_1，e_2，e_3都是割边。

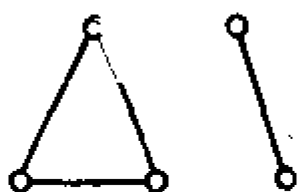


图 2.16

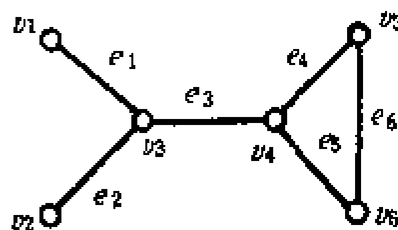


图 2.17

定理 2.9 在无向图G中，对结点a，当且仅当存在两个结点u，v，它们之间的所有通路均通过点a时，a才是割点。

证：如果a是割点，则去掉a及其关联边后图将不连通，在两个不连通的分支中各取一个结点u与v，显然u与v不连通，必须经过a才能形成通路。即如果a是割点，图中必然存在这样的两个结点u与v。

反之，如果存在两个结点u与v必须经过a才能构成通路，则去掉a及其关联边后u与v将不通，即图成为非连通图，根据定义，a是图的割点。 ▮

如果图不存在割点，称图为不可分图。

定理 2.10 当且仅当无向图G的一条边e不包含在回路中时，e才是割边。

证： 设e是割边，它的两个关联点为v_i与v_j，如果e是回路的边，则去掉e后，v^i与v_j仍连通，与e是割边的定义不符，因此e不可能是回路的一条边。

反之，如果e不是任何回路的一条边，则e的两个关联点v_i与v_j只有经过e才能连通，因此去掉e后，v_i与v_j即不连通，图成为非连通图，根据定义，e是割边。 ▌

连通无向图还有以下两个常用到的性质：

1）若连通无向图G的子图G'不含G的全部结点，则G中一定存在不属于G'的一条边，它的一个端点在G'内而另一个端点不在G'内。

2）若连通图G的子图G'不含G的所有边，则一定可以找到一条边，它不属于G'，但有一个端点在G'内。

两个性质是显然的，证明从略。

§2.3 图的矩阵表示

前面我们讲了用图形表示图的方法，它的优点是直观、形象，便于理解，但如果图比较复杂，从图形上去分析图也是不方便的，一般都用矩阵表示图，这种方法简单，使用方便，特别是它将图的问题变成了矩阵运算的问题，更适于用计算机进行计算或处理，所以矩阵法是分析研究图的一个最有力的工具。由于研究问题的出发点不同，图的矩阵表示也有多种形式，这一节我们主要介绍三种形式的矩阵，即邻接矩阵、可达性矩阵与关联矩阵。

一、邻接矩阵

定义 2.21 （无向图的邻接矩阵）设$G=(V,E)$是一个简单无向图，$V=\{v_1,v_2,\cdots,v_n\}$，则G的邻接矩阵$A=(a_{ij})$是一个n阶方阵，其中

$$a_{ij}=\begin{cases}1\text{，若 } v_i \text{ 与 } v_j \text{ 邻接}\\ 0\text{，否则}\end{cases}$$

图 2.18 的邻接矩阵列于其左

$$A=\begin{matrix} & v_1 & v_2 & v_3 & v_4 & v_5 \\ v_1 & 0 & 1 & 1 & 1 & 1 \\ v_2 & 1 & 0 & 1 & 0 & 0 \\ v_3 & 1 & 1 & 0 & 1 & 0 \\ v_4 & 1 & 0 & 1 & 0 & 1 \\ v_5 & 1 & 0 & 0 & 1 & 0 \end{matrix}.$$

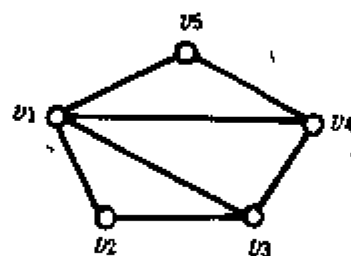


图 2.18

由定义可知：无向图的邻接矩阵是一个对称方阵，它的对角线元素全为“0”(因为是简单图无自环)，每一行（列)中“1”的个数是对应结点的次数。如矩阵的所有元素全为“0”，对应的是零图；除对角线外如所有元素全为“1”，则对应的是完全图。如果改变结点的排列次序，相当于矩阵的行、列置换，矩阵的性质是不变的，因而所表示的图是不变的，所以我们可以选定结点的一种排列次序，由它得到的邻接矩阵作为图的邻接矩阵。

这也说明，如果 G_2 的邻接矩阵 A_2，可以通过 G_1 的邻接矩阵 A_1 的行（列）置换得到，则图 G_2 与 G_1 是同构的。

当且仅当图 G 的邻接矩阵 A 可以分成如下的块状形式时，图 G 是一个有两个连通分支

$$A=\begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}$$

G_1，G_2 的非连通图。其中 A_{11} 是对应子图 G_1 的邻接矩阵，A_{22} 是对应子图 G_2 的邻接矩阵。

定义 2.22 元素只有“0”或“1”的矩阵称为布尔矩阵。

定义 2.23 （有向图的邻接矩阵）设 $D=(V,E)$ 是一个简单有向图，$V=\{v_1, v_2, \cdots, v_n\}$，则 D 的邻接矩阵 $A=(a_{ij})$ 是一个 n 阶方阵，其中

$$a_{ij}=\begin{cases} 1, & 如\langle v_i, v_j\rangle \in E \\ 0, & 否则 \end{cases}$$

图 2.19 的邻接矩阵列于其右。

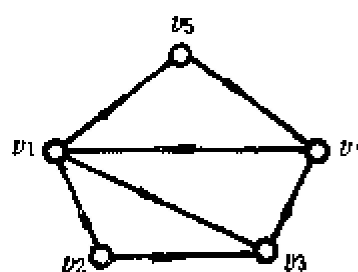


图 2.19

$$A=\begin{matrix} & v_1 & v_2 & v_3 & v_4 & v_5 \\ v_1 & 0 & 0 & 1 & 0 & 1 \\ v_2 & 1 & 0 & 1 & 0 & 0 \\ v_3 & 0 & 0 & 0 & 1 & 0 \\ v_4 & 1 & 0 & 0 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 1 & 0 \end{matrix}$$

有向图的邻接矩阵与无向图的邻接矩阵都是 n 阶布尔矩阵，但有向图的邻接矩阵不一定对称，第 i 行中“1”的数目表示结点 v_i 的引出次数，第 j 列中“1”的数目表示结点 v_j 的引入次数。

有向图的一些特性，可以通过邻接矩阵的运算而得，现就矩阵的三种运算分析如下。

（1）矩阵 AA^T

设 A 是有向图的邻接矩阵，A^T 是 A 的转置矩阵，$A=(a_{ij})$，$A^T=(a'_{ij})=(a_{ji})$，定义

$$B=AA^T=(b_{ij})$$

其中

$$b_{ij}=\sum_{k=1}^{n} a_{ik}a'_{kj}=\sum_{k=1}^{n} a_{ik}a_{jk}$$
$$=a_{i1}a_{j1}+a_{i2}a_{j2}+\cdots+a_{ik}a_{jk}+\cdots+a_{in}a_{jn}$$

根据邻接矩阵的定义可知，如 $a_{ik}=1$ 表示有一条弧从结点 v_i 引向 v_k。只有 $a_{ik}=1$ 且 $a_{jk}=1$ 才会有 $a_{ik}a_{jk}=1$，故 $a_{ik}a_{jk}=1$ 表示从 v_i 和 v_j 各引一条弧终止于 v_k，b_{ij} 的每一项只有“0”和“1”两种可能，所以 b_{ij} 的数值等于为“1”的项数，因此 b_{ij} 的值表示从结点 v_i 和 v_j 各自引出的弧终止于同一结点的结点数。同理 b_{ii} 的值表示从结点 v_i 引出弧的数目，即 v_i 的引出次数。由式可知 $b_{ij}=b_{ji}$，所以矩阵 B 是一个对称矩阵。

例 2.7 设有向图 D 的图形如图 2.20 所示，求 $B=AA^T$

解：

$$A=\begin{matrix} & v_1 & v_2 & v_3 & v_4 \\ v_1 & 0 & 0 & 0 & 1 \\ v_2 & 1 & 0 & 0 & 1 \\ v_3 & 1 & 1 & 0 & 1 \\ v_4 & 0 & 0 & 1 & 0 \end{matrix} \quad A^T=\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

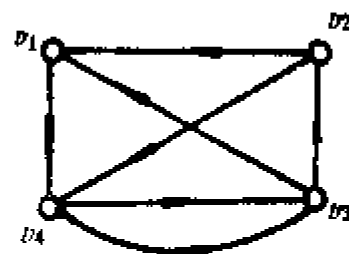


图 2.20

$$B=AA^{T}=\begin{pmatrix}0&0&0&1\\1&0&0&1\\1&1&0&1\\0&0&1&0\end{pmatrix}\times\begin{pmatrix}0&1&1&0\\0&0&1&0\\0&0&0&1\\1&1&1&0\end{pmatrix}=\begin{pmatrix}1&1&1&0\\1&2&2&0\\1&2&3&0\\0&0&0&1\end{pmatrix}$$

从矩阵B的元素值可得出如下结论：

$b_{14}=0$ 表示从结点 v_1 与 v_4 所引出的弧没有共同的终止结点。

$b_{23}=2$ 表示从 v_2 与 v_3 引出的弧共同终止的结点有 2 个。(图中可看出是 v_1 与 v_4)

$b_{33}=3$ 表示 v_3 的引出次数为 3 。

⋮

这些结论与图的图形完全吻合。

在求B的运算中，可以不写出 A^T，只从A求B，方法是 b_{ij} 等于A的第 i 行与第 j 行对应元素乘积之和。

（2）矩阵 A^TA

定义 2.24 $\bar{B}=A^TA=(\bar{b}_{ij})$，其中

$$\bar{b}_{ij}=\sum_{k=1}^{n}a'_{ik}a_{kj}=\sum_{k=1}^{n}a_{ki}a_{kj}$$
$$=a_{1i}a_{1j}+a_{2i}a_{2j}+\cdots+a_{ki}a_{kj}+\cdots+a_{ni}a_{nj}$$

与对 b_{ij} 的分析类似，我们也可得出：$\bar{b}_{ij}$ 的值表示有引出弧同时引向结点 v_i 与 v_j 的那些结点的数目，而 $\bar{b}_{ii}$ 的数值则表示结点 v_i 的引入次数，同样 $\bar{B}$ 也是对称于对角线的方阵。在求 $\bar{B}$ 的运算中，也可不写出 A^T，只从A求 $\bar{B}$，方法是 $\bar{b}_{ij}$ 等于A的第 i 列与第 j 列对应元素乘积之和。

例 2.8 如例 2.7，可得

$$\bar{B}=\begin{pmatrix}0&1&1&0\\0&0&1&0\\0&0&0&1\\1&1&1&0\end{pmatrix}\times\begin{pmatrix}0&0&0&1\\1&0&0&1\\1&1&0&1\\0&0&1&0\end{pmatrix}=\begin{pmatrix}2&1&0&2\\1&1&0&1\\0&0&1&0\\2&1&0&3\end{pmatrix}$$

其中$\bar{b}_{14}=2$ 表示有两个结点它们都有引出弧终止于结点 v_1 与 v_4。$\bar{b}_{44}=3$ 表示 v_4 的引入次数为 3，这些结论与图的图形完全吻合。

（3）矩阵的幂 A^m

先看矩阵的二次幂 $A^2=(a_{ij}^{(2)})$，其中

$$a_{ij}^{(2)}=\sum_{k=1}^{n}a_{ik}a_{kj}=a_{i1}a_{1j}+a_{i2}a_{2j}+\cdots+a_{in}a_{nj}$$

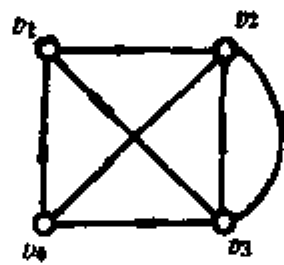


图 2.21

当且仅当 $a_{ik}=1$ 且 $a_{kj}=1$ 时才有 $a_{ik}a_{kj}=1$，故 $a_{ik}a_{kj}=1$ 表示有一条从结点 v_i 出发经 v_k 而终止于 v_j 的长度为2的路径，所以 $a_{ij}^{(2)}$ 的值表示从结点 v_i 到 v_j 长度为 2 的路径数目，而 $a_{ii}^{(2)}$ 则表示从 v_i 引出又回到 v_i 的长度为 2 的回路的数目。

例 2.9 设有向图D的图形如图 2.21 所示，则

$$A^2=AA=\begin{pmatrix}0&1&0&0\\0&0&1&1\\1&1&0&1\\1&0&0&0\end{pmatrix}\times\begin{pmatrix}0&1&0&0\\0&0&1&1\\1&1&0&1\\1&0&0&0\end{pmatrix}=\begin{pmatrix}0&0&1&1\\2&1&0&1\\1&1&1&1\\0&1&0&1\end{pmatrix}$$

其中 $a_{21}^{(2)}=2$，表示从结点 v_2 到 v_1 长度为 2 的路径有两条，从图形上可以看到的确存在这样两条路径，它们是$v_2\to v_3\to v_1$及 $v_2\to v_4\to v_1$。又 $a_{23}^{(2)}=0$，表明从结点 v_2 到 v_3 不存在长度为 2 的路径，$a_{33}^{(2)}=1$，表明 v_3 上长度为 2 的回路只有一条，从图形上可以看出这条回路是 $v_3\to v_2\to v_3$。矩阵运算得出的结论与图形是吻合的。由此可推广到一般m次幂得到下面定理。

定理 2.11 设$D=(V,E)$是具有结点集合$V=\{v_1, v_2,\cdots,v_n\}$的简单有向图，$A$是$D$的邻接矩阵，则$A^m(m=1, 2,\cdots)$中的$a_{ij}^{(m)}$值等于从 v_i 到 v_j 长度为m的路径数目。

证：对m用归纳法证明，设$m=1$则$A^m=A$，按定义结论成立，当$m=2$时，上面已分析结论成立，设$m=r$时结论成立，因

$$A^{r+1}=A^rA=(a_{ij}^{(r+1)})，其中$$

$$a_{ij}^{(r+1)}=\sum_{k=1}^{n}(a_{ik}^{(r)}\cdot a_{kj})$$

当 $a_{ik}^{(r)}$ 与 a_{kj} 不为 0 时，前者表示从 v_i 到 v_k 长度为 r 的路径数目，后者表示从 v_k 到 v_j 长度为 1 的路径，所以 $a_{ik}^{(r)}\cdot a_{kj}$表示从 v_i 经 v_k 到 v_j 长度为 $(r+1)$ 的路径的数目，对所有k求和即得 $a_{ij}^{(r+1)}$，它就是从 v_i 到 v_j 长度为 $(r+1)$ 的路径总数，故当$m=r+1$时结论亦成立。 ▮

例 2.10 对图 2.21 所表示的有向图，可求出 A^3，A^4如下：

$$A^3=\begin{pmatrix}2&1&0&1\\1&2&1&1\\2&2&1&2\\0&0&1&1\end{pmatrix},\quad A^4=\begin{pmatrix}1&2&1&1\\2&2&2&3\\3&3&2&3\\2&1&0&1\end{pmatrix}$$

从 A^3 可得：$a_{14}^{(3)}=1$，表示从 v_1 到 v_4 有而且只有一条长度为 3 的路径。(从图形可看出是 $v_1\to v_2\to v_3\to v_4$)，$a_{42}^{(3)}=0$，表示从 v_4 到 v_2 不存在长度为 3 的路径，$a_{22}^{(3)}=2$，表示v_2 上有两条长度为 3 的回路。如此等等。

从 A^4 可得：$a_{12}^{(4)}=2$，表示从 v_1 到 v_2 有两条长度为 4 的路径，$a_{33}^{(4)}=2$，表示v_3 上有两条长度为 4 的回路，如此等等。

求不同长度的路径或回路，如果从图形上去找，不仅不容易，还有可能遗漏，而用矩阵算法是准确无误的，而且执行矩阵运算，对计算机来说并非难事。当然，如果不仅要求路径的长度，还要找出路径所经的结点时，上面的算法就不够了。

定义 2.25 （带权有向图的邻接矩阵）设$D=(V,E)$，是一简单带权有向图，$V=\{v_1, v_2,\cdots,v_n\}$，其对应的邻接矩阵$A=(a_{ij})$，其中

$$a_{ij}=\begin{cases}w，若\langle v_i, v_j\rangle\in E 且 w 是它的权\\0，若 i=j\\\infty，若\langle v_i, v_j\rangle\notin E\end{cases}$$

图 2.22 所示的带权有向图的邻接矩阵列于其左。

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 2 & 3 & \infty \\ 1 & 0 & 5 & 8 \\ \infty & 4 & 0 & \infty \\ 1 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

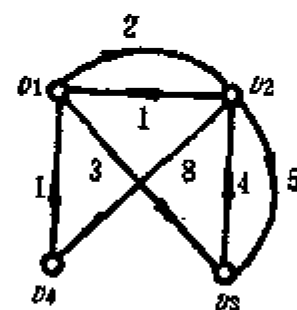


图 2.22

对于带权无向图的邻接矩阵的定义是类似的，不再赘述。

二、可达性矩阵

定义 2.26 设$D=(V,E)$是一简单有向图，$V=\{v_1, v_2,\cdots,v_n\}$，$D$的可达性矩阵$P=(p_{ij})$是一$n$阶方程，其中

$$p_{ij}=\begin{cases} 1, & 如果从\ v_i\ 可到达\ v_j \\ 0, & 否则 \end{cases}$$

可达性矩阵仅考虑图中任意两个结点之间是否可到达以及任意结点上是否存在回路的问题，对于路径的长度及数目是不予考虑的。可达性矩阵也称路径矩阵或道路矩阵。

图 2.23 的有向图的可达性矩阵列于其左。

$$P = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

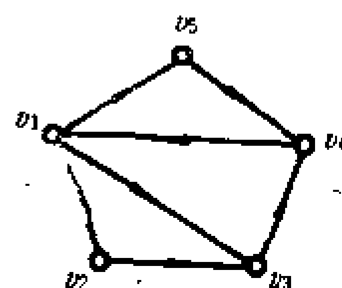


图 2.23

根据图的图形写出它的可达性矩阵并不是我们的目的，因为它仍然脱离不开对图形的依赖性。我们的目的是要寻求一种算法，可以不依赖图形仍能对图的可达性问题作出判断。下面介绍几种算法。

因为图的邻接矩阵及其幂矩阵也是描述结点路径的，我们可以利用它们求出图的可达性矩阵，例如先考察邻接矩阵A，如果$a_{ij}=1$，显然从v_i可到达v_j（路径长度为1），于是即得$p_{ij}=1$。如果$a_{ij}=0$，表明v_i至v_j不存在长度为1的路径，但还不能作出v_i不能到达v_j的结论，因为可能存在长度大于1的路径，因此必须考察A^2，如果$a_{ij}^{(2)}\neq 0$，从v_i也可到达v_j（路径长度为2），于是$p_{ij}=1$。如果$a_{ij}^{(2)}=0$，可再考察A^3，如此继续下去，由于图是有限的，根据定理 2.5 和定理 2.4，如果从v_i可到达v_j的话，一定存在一条基本路径，它的长度小于等于$(n-1)$，如是基本回路则长度小于等于n。因此我们只要查到A^n就可以了，如从a_{ij}直到$a_{ij}^{(n)}$都等于0，则$p_{ij}=0$，否则$p_{ij}=1$。根据以上分析，我们归纳出从邻接矩阵求可达性矩阵的算法步骤如下

1）求出D的邻接矩阵的各次幂A，$A^2,\cdots,A^n$。

2）令$Q=A+A^2+\cdots+A^n=(q_{ij})$，其中

$$q_{ij}=a_{ij}+a_{ij}^{(2)}+\cdots+a_{ij}^{(n)}$$

3）将Q化为P，方法是如$q_{ij}\neq 0$，则$p_{ij}=1$，如$q_{ij}=0$，则$p_{ij}=0$。

例 2.11 如图 2.23 的有向图，求P的步骤如下：

$$A=\begin{matrix} & v_1 & v_2 & v_3 & v_4 & v_5 \\ v_1 & 0 & 0 & 1 & 0 & 1 \\ v_2 & 1 & 0 & 1 & 0 & 0 \\ v_3 & 0 & 0 & 0 & 1 & 0 \\ v_4 & 1 & 0 & 0 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 1 & 0 \end{matrix}\qquad A^2=\begin{pmatrix} 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$A^3=\begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}\qquad A^4=\begin{pmatrix} 0 & 0 & 2 & 0 & 2 \\ 2 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \end{pmatrix}$$

$$A^5=\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 2 & 2 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 2 \\ 2 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$Q=A+A^2+\cdots+A^5=\begin{pmatrix} 2 & 0 & 3 & 6 & 3 \\ 4 & 0 & 5 & 5 & 4 \\ 3 & 0 & 1 & 3 & 1 \\ 3 & 0 & 3 & 2 & 3 \\ 3 & 0 & 1 & 3 & 1 \end{pmatrix},\ 故 P=\begin{pmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

上面的算法是可行的，但计算较繁琐。可以考虑简化算法。首先可达性矩阵是一个布尔矩阵，而在求Q的运算过程中，是非布尔矩阵的相加，在求A的各次幂A^m时，也是非布尔矩阵的运算，最后求P时再化为布尔矩阵，这就说明前面那些非布尔矩阵运算所得到的非零数字是毫无意义的，反而增加运算的复杂性。这就启发我们，可以采用矩阵的布尔运算来求可达性矩阵。为此，先介绍布尔乘与布尔和的概念。

用符号“∧”表示布尔乘，它的定义为：$0\wedge0=0$，$0\wedge1=0$，$1\wedge0=0$，$1\wedge1=1$。用符号“∨”表示布尔和，它的定义为：$0\vee0=0$，$0\vee1=1$，$1\vee0=1$，$1\vee1=1$。

布尔矩阵A与B的和，记作$A\vee B$，结果为布尔矩阵$C=A\vee B=(c_{ij})$，其中

$$c_{ij}=a_{ij}\vee b_{ij}$$

布尔矩阵A与B的乘积记作$A\otimes B$，结果为布尔矩阵$D=A\otimes B=(d_{ij})$，其中

$$d_{ij}=\bigvee_{k=1}^{n}(a_{ik}\wedge b_{kj})$$

有了上面的定义，布尔矩阵的布尔运算记为：

$$A^{(2)}=A\otimes A,\ A^{(3)}=A^{(2)}\otimes A,\cdots,A^{(n)}=A^{(n-1)}\otimes A$$

采用矩阵的布尔运算，求可达性矩阵的计算步骤如下：

1）求出有向图D的邻接矩阵的各次幂的布尔形式A，$A^{(2)},\cdots,A^{(n)}$。

2）用矩阵的布尔和即可求出$P=(p_{ij})$

$$P = A \vee A^{(2)} \vee \cdots \vee A^{(n)} = \bigvee_{k=1}^{n} A^{(k)}$$

例 2.11 用矩阵的布尔算法如下：

$$A^{(2)} = A \otimes A = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

同理可求出

$$A^{(3)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}, \quad A^{4} = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, A^{(5)} = A^{(2)}$$

故可达性矩阵为

$$P = A \vee A^{(2)} \vee \cdots \vee A^{(5)} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

可见，用矩阵的布尔运算比一般的矩阵运算更简单些，但当图的结点较多时，计算起来仍然是很麻烦的。

1962年，Warshall 提出一个有效算法，直接从邻接矩阵求可达性矩阵，它是一个比较简单的程序，在计算机上是很容易实现的，算法步骤如下：

（1）置新矩阵 $P := A$

（2）置 $i := 1$

（3）对所有的 j，如果 $P(j,i) = 1$，则对 $k = 1, 2 \cdots n$

$$P(j,k) := P(j,k) \vee P(i,k)$$

（4） $i := i + 1$

（5）如 $i \leqslant n$ 转到步骤（3），否则停止。

由算法步骤可知，Warshall 算法须经历 i，j，k 三次循环，故算法的时间复杂性为 $0(n^3)$。

例 2.12 用 Warshall 算法求例 2.11 的可达性矩阵步骤如下：

（1）置新矩阵

$$P:=A=\begin{bmatrix}0&0&1&0&1\\1&0&1&0&0\\0&0&0&1&0\\1&0&0&0&0\\0&0&0&1&0\end{bmatrix}$$

（2）置 $i:=1$

（3）考察所有的 j，当 $j=1$ 时 $P(j,i)=P(1,1)=0$，不作处理，继而看 $j=2$，$P(2,1)=1$，于是将 P 中第 2 行与第 1 行的对应元素进行布尔和，结果记入第二行的对应位置，于是得

$$P:=\begin{bmatrix}0&0&1&0&1\\1&0&1&0&1\\0&0&0&1&0\\1&0&0&0&0\\0&0&0&1&0\end{bmatrix}$$

再看 $j=3$，因 $P(j,i)=P(3,1)=0$，不作处理，继而看 $j=4$，此时 $P(j,i)=P(4,1)=1$，于是将 P 中第 4 行与第 1 行对应元素进行布尔和，结果记入第 4 行的对应位置，于是得

$$P:=\begin{bmatrix}0&0&1&0&1\\1&0&1&0&1\\0&0&0&1&0\\1&0&1&0&1\\0&0&0&1&0\end{bmatrix}$$

再看 $j=5$，因 $P(j,i)=P(5,1)=0$，不作处理，至此所有的 j 都已察遍，转入下一步。

（4）$i:=i-1=1+1=2$

（5）因 $i\leqslant n(5)$，转回到步骤（3）

（3）考察所有的 j，均有 $P(j,i)=P(j,z)=0$，因此这一循环结束，P 不变，转入（4）

（4）$i:=i+1=2+1=3$

（5）因 $i\leqslant n(5)$，转回到步骤（3）

（3）考察所有的 j，当 $j=1,2,4$ 时 $P(j,i)=1$，故依次将第 1 行与第 3 行、第 2 行与第 3 行、第 4 行与第 3 行对应元素进行布尔和，结果分别记入第 1、2、4 行，于是得

$$P:=\begin{bmatrix}0&0&1&1&1\\1&0&1&1&1\\0&0&0&1&0\\1&0&1&1&1\\0&0&0&1&0\end{bmatrix}$$

对所有的 j 已察遍，转入步骤（4）

（4）$i:=i+1=3+1=4$

（5）因 $i\leqslant n$，转回到步骤（3）

（3）对所有的 j 都有 $P(i,\ j)=P(j,\ 4)=1$，

因此每一行都与第4行的对应元素进行布尔和，结果记入各行，于是得

$$P:=\begin{bmatrix}1&0&1&1&1\\1&0&1&1&1\\1&0&1&1&1\\1&0&1&1&1\\1&0&1&1&1\end{bmatrix}$$

对所有的 j 已察遍，转入步骤(4)

(4) $i:=i+1=4+1=5$

(5) 因 $i\leqslant n$，转回到步骤(3)

(3) 对所有的 j 都有 $P(j,i)=P(j,5)=1$，因此每一行都与第5行的对应元素进行布尔和，结果记入各行，于是得

$$P:=\begin{bmatrix}1&0&1&1&1\\1&0&1&1&1\\1&0&1&1&1\\1&0&1&1&1\\1&0&1&1&1\end{bmatrix}$$

转入下一步时，$i=6>n(5)$，计算结束，结果 P 即是所求的可达性矩阵。

图的连通性，可以用可达性矩阵进行判断。

定理 2.12 设 D 是简单有向图，A，P 是 D 的邻接矩阵与可达性矩阵，A^T，P^T 分别是它们的转置矩阵，则

1) D 是强连通的当且仅当 P 的元素全为1。

2) D 是单向连通的当且仅当 P 与 P^T 的布尔和 $P'=P\vee P^T$ 除对角线元素外全为1。

3) D 是弱连通的当且仅当 A 与 A^T 的布尔和 $A'=A\vee A^T$ 的可达性矩阵的元素全为1。

根据邻接矩阵与可达性矩阵的意义，上面的判断条件是不难理解的，这里不作证明，只举例说明如下。

例 2.13 设四个有向图 D_1，D_2，D_3，D_4 的邻接矩阵分别为下面的 A_1,A_2,A_3，A_4，试判断图的连通性。

$$A_1=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix},\quad A_2=\begin{bmatrix}0&1&0\\0&0&0\\1&0&0\end{bmatrix},\quad A_3=\begin{bmatrix}0&1&1\\0&0&0\\0&0&0\end{bmatrix},\quad A_4=\begin{bmatrix}0&0&0\\0&0&0\\1&0&0\end{bmatrix}$$

解：D_1 的可达性矩阵为

$$P_1=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

故 D_1 是强连通的。D_2 的可达性矩阵为

$$P_2=\begin{bmatrix}0&1&0\\0&0&0\\1&1&0\end{bmatrix}$$

故 D_2 不是强连通的，进而求 P_2^T 及 P_2'

$$P'_2=P_2\vee P_2^T=\begin{pmatrix}0&1&0\\0&0&0\\1&1&0\end{pmatrix}\vee\begin{pmatrix}0&0&1\\1&0&1\\0&0&0\end{pmatrix}=\begin{pmatrix}0&1&1\\1&0&1\\1&1&0\end{pmatrix}$$

故 D_2 是单向连通的。D_3 的可达性矩阵为

$$P_3=\begin{pmatrix}0&1&1\\0&0&0\\0&0&0\end{pmatrix}$$

故 D_3 不是强连通的，进而求 P_3^T 及 P'_3

$$P'_3=P_3\vee P_3^T=\begin{pmatrix}0&1&1\\0&0&0\\0&0&0\end{pmatrix}\vee\begin{pmatrix}0&0&0\\1&0&0\\1&0&0\end{pmatrix}=\begin{pmatrix}0&1&1\\1&0&0\\1&0&0\end{pmatrix}$$

故 D_3 也不是单向连通的，转而求 A_3^T 及 A'_3

$$A'_3=A_3\vee A_3^T=\begin{pmatrix}0&1&1\\0&0&0\\0&0&0\end{pmatrix}\vee\begin{pmatrix}0&0&0\\1&0&0\\1&0&0\end{pmatrix}=\begin{pmatrix}0&1&1\\1&0&0\\1&0&0\end{pmatrix}$$

再求 A'_3 的可达性矩阵 P''_3

$$P''_3=\begin{pmatrix}1&1&1\\1&1&1\\1&1&1\end{pmatrix}$$

故 D_3 是弱连通的。D_4 的可达性矩阵为

$$P_4=\begin{pmatrix}0&0&0\\0&0&0\\1&0&0\end{pmatrix}$$

故 D_4 不是强连通的，进而求 P_4^T 及 P'_4

$$P'_4=P_4\vee P_4^T=\begin{pmatrix}0&0&0\\0&0&0\\1&0&0\end{pmatrix}\vee\begin{pmatrix}0&0&1\\0&0&0\\0&0&0\end{pmatrix}=\begin{pmatrix}0&0&1\\0&0&0\\1&0&0\end{pmatrix}$$

故 D_4 也不是单向连通的，转而求 A_4^T 及 A'_4

$$A'_4=A_4\vee A_4^T=\begin{pmatrix}0&0&0\\0&0&0\\1&0&0\end{pmatrix}\vee\begin{pmatrix}0&0&1\\0&0&0\\0&0&0\end{pmatrix}=\begin{pmatrix}0&0&1\\0&0&0\\1&0&0\end{pmatrix}$$

再求 A'_4 的可达性矩阵 P''_4

$$P''_4=\begin{pmatrix}1&0&1\\0&0&0\\1&0&1\end{pmatrix}$$

故 D_4 也不是弱连通的，所以 D_4 是非连通图。

将这四个有向图的图形画出来，如图 2.24，可见判断是正确的。

下面定理，给出了如何利用可达性矩阵，求出图的所有强分图。

定理 2.13 设有向图 D 的可达性矩阵为 P，P^T 是 P 的转置矩阵，定义矩阵 $Q=P\odot$

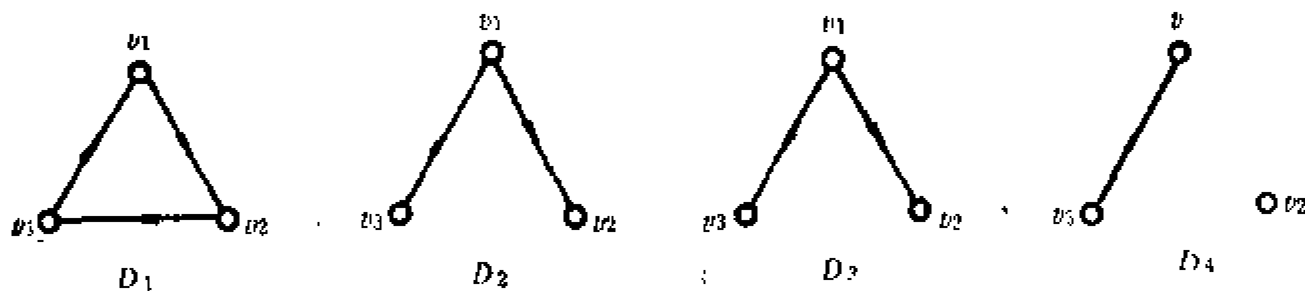


图 2.24

$P^T=(q_{ij})$。其中

$$q_{ij}=p_{ij}\cdot p^T_{ij}=p_{ij}\cdot p_{ji}\text{（“·”为普通乘法运算）}$$

则包含结点 v_i 的强分图 D_i 是由 Q 的第 i 行元素为 1 的对应结点所组成。

证：设任一结点 v_j，如 Q 中有 $q_{ij}=1$，则必有 $p_{ij}=1$ 及 $p_{ji}=1$，说明结点 v_i 与 v_j 是可以互相到达的，故 v_j 必与 v_i 处于同一个强分图中，同理如有 $q_{ik}=1$，则 v_k 亦与 v_i 处于同一个强分图中，因此，与 v_i 处于同一个强分图的结点必然是 Q 中第 i 行元素为 1 所对应的结点。 ▮

例 2.14 已知有向图 D 的可达性矩阵 P 如下，求 D 的所有强分图。

$$P=\begin{bmatrix}0&1&0&1&1\\0&1&0&1&1\\1&1&0&1&1\\0&1&0&1&1\\0&1&0&1&1\end{bmatrix}$$

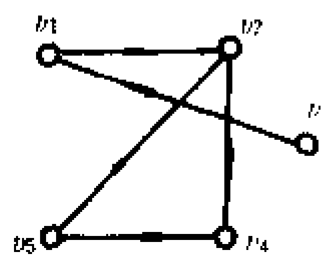


图 2.25

解：先求 P^T、继而求出 Q 如下

$$Q=P\odot P^T=\begin{bmatrix}0&1&0&1&1\\0&1&0&1&1\\1&1&0&1&1\\0&1&0&1&1\\0&1&0&1&1\end{bmatrix}\odot\begin{bmatrix}0&0&1&0&0\\1&1&1&1&1\\0&0&0&0&0\\1&1&1&1&1\\1&1&1&1&1\end{bmatrix}$$

$$=\begin{bmatrix}0&0&0&0&0\\0&1&0&1&1\\0&0&0&0&0\\0&1&0&1&1\\0&1&0&1&1\end{bmatrix}$$

从 Q 可以看出，没有任何一个结点与 v_1 及 v_3 是可互相到达的，故 v_1 及 v_3 必自成一强分图，与 v_2 可互相到达的结点是 v_4 与 v_5，因此 D 的所有强分图是：$\{v_1\}$，$\{v_3\}$，$\{v_2,v_4,v_5\}$。图 2.25是这个图的图形，可见判断是正确的。

以上是用可达性矩阵对有向图的连通性进行判断，关于无向图的连通性判断，在 § 3.8 中再作详细讨论。

三、关联矩阵

定义 2.27 （无向图的完全关联矩阵）设 $G=(V,E)$ 是一个简单无向图，$V=\{v_1,$

$v_2,\cdots,v_n\}$，$E=\{e_1, e_2,\cdots,e_m\}$，令$B=(b_{ij})_{n\times m}$，其中

$$b_{ij}=\begin{cases}1\text{，若 } v_i \text{ 关联 } e_j\\ 0\text{，否则}\end{cases}$$

则称B为无向图G的完全关联矩阵。

例 2.15 图 2.26 所表示的无向图的完全关联矩阵如下

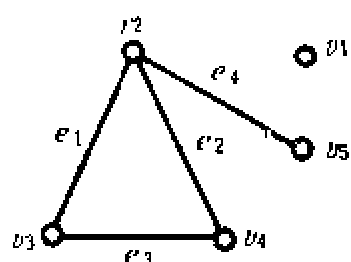


图 2.26

$$B=\begin{matrix} & e_1 & e_2 & e_3 & e_4 \\ v_1 & 0 & 0 & 0 & 0 \\ v_2 & 1 & 1 & 0 & 1 \\ v_3 & 1 & 0 & 1 & 0 \\ v_4 & 0 & 1 & 1 & 0 \\ v_5 & 0 & 0 & 0 & 1\end{matrix}$$

从定义及以上例子可知，完全关联矩阵每一列中只有两个“1”，每一行“1”的数目是该行对应结点的次数，完全为“0”的行对应的结点是孤点，只有一个“1”的行，“1”所对应的边是悬挂边，它不在图的任何回路内，变换图的结点或边的次序时，得到的矩阵是置换等价的，完全关联矩阵也是一个布尔矩阵。

当且仅当矩阵可分成如下的块状形式时，图G是一个有两个连通分支的非连通图。

$$B=\begin{bmatrix} B_{11} & \bigcirc \\ \bigcirc & B_{22}\end{bmatrix}$$

定义 2.28 (有向图的完全关联矩阵)设$D=(V,E)$是一简单有向图,$V=\{v_1,v_2,\cdots,v_n\}$，$E=\{e_1, e_2,\cdots,e_m\}$，令$B=(b_{ij})_{n\times m}$，其中

$$b_{ij}=\begin{cases}1\text{，若 } v_i \text{ 是 } e_j \text{ 的起点}\\ -1\text{，若 } v_i \text{ 是 } e_j \text{ 的终点}\\ 0\text{，否则}\end{cases}$$

则称B为有向图D的完全关联矩阵。

例 2.16 图 2.27 所表示的有向图的完全关联矩阵如下：

$$B=\begin{matrix} & e_1 & e_2 & e_3 & e_4 & e_5 \\ v_1 & 1 & 0 & 0 & 0 & 0 \\ v_2 & -1 & 1 & 0 & 0 & -1 \\ v_3 & 0 & -1 & 1 & -1 & 0 \\ v_4 & 0 & 0 & -1 & 1 & 1 \\ v_5 & 0 & 0 & 0 & 0 & 0\end{matrix}$$

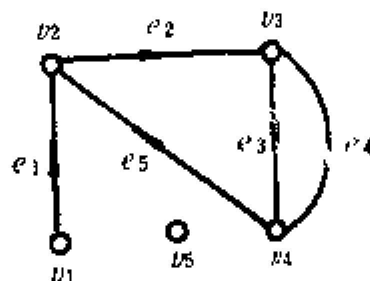


图 2.27

从定义及以上例子可知，有向图的完全关联矩阵每一列之和为零，每一行“1”的数目是该行对应结点的引出次数，“−1”的数目是对应结点的引入次数，完全为“0”的行对应的结点是孤点，其余性质与无向图类似。

在线性代数中，一个矩阵线性无关的行的数目，称为该矩阵的秩，这一定义也适用于完全关联矩阵，我们知道，矩阵的行相加，是矩阵的一种初等变换，矩阵的特性是不变的。但是在完全关联矩阵中作行相加时，可能会出现$1+1=2$的情况，结果将不再符合完全关联矩阵的定义，为了保证运算的封闭性，我们作出如下规定：对无向图，行相加采用模2加法（环和），即$0\oplus 0=0$，$0\oplus 1=1\oplus 0=1$，$1\oplus 1=0$。对有向图，行相加即

为普通加法运算。

下面以无向图的完全关联矩阵为例进行分析，得出的结论，对有向图的完全关联矩阵也是适用的。

例 2.17 图 2.28 的完全关联矩阵如下：

$$B=\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array}\begin{array}{c} \begin{array}{ccccccc} a & b & c & d & e & f & g \end{array} \\ \left[\begin{array}{ccccccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}\right] \end{array}$$

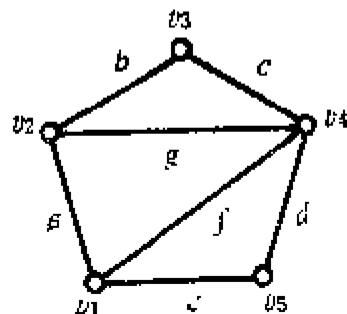


图 2.28

把矩阵的每一行看作是一个行向量 a_i，则整个行向量的环和为零，即

$$a_1 \oplus a_2 \oplus \cdots \oplus a_5 = 0$$

因此这个行向量组是线性相关的，其中任一行向量均可由其余的行向量线性表出，例如

$$a_5 = a_1 \oplus a_2 \oplus a_3 \oplus a_4$$

这一特性亦适用于有向图，因此得出如下引理

引理一 一个连通图的完全关联矩阵 B 最多有 $(n-1)$ 个独立行。(n 为图的结点数)

就是说在 B 的行向量中，最少有一行是非独立的。但是还有下面一条引理

引理二 一个连通图的完全关联矩阵 B 最少有 $(n-1)$ 个独立行。

证：用反证法，设 B 的独立行数少于 $(n-1)$，那么 $(n-1)$ 个行向量组一定是线性相关的，即

$$k_1a_1 \oplus k_2a_2 \oplus \cdots \oplus k_{n-1}a_{n-1} = 0$$

其中 $k_1, \cdots, k_{n-1}$ 是不全为 0 的常数（这里只取 0 或 1），上式表明 $(n-1)$ 个行向量的环和为零，这是不可能的，因为 B 的任何一行中最少有一个 1（否则该行对应的点为弧点，与连通图的前提矛盾），而每一列中必须有两个 1，现在假设第 i 行中有一个 1 位于第 j 列，第 j 列的另一个 1 在第 k 行中，现在去掉第 i 行，把其余的 $(n-1)$ 个行进行环和，显然第 j 列的结果必为 1 而不是 0，就是说 $(n-1)$ 个行向量环和的结果不可能是 0 向量，与上面假设矛盾，因此 B 最少有 $(n-1)$ 个独立行。 ▮

有了以上两条引理，即可得出如下定理。

定理 2.14 具有 n 个结点的连通图的完全关联矩阵的秩是 $R(B)=n-1$。

因此在完全关联矩阵中，任意去掉一行仍能保持矩阵的特性，于是引出下面关联矩阵的定义。

定义 2.29 在有 n 个结点 m 条边的连通图的完全关联矩阵中，去掉任一行后得到的 $(n-1)\times m$ 矩阵，称为图的关联矩阵，去掉的行所对应的结点称为参考点。

在图的矩阵表示中，经常用到的是图的关联矩阵。

如例2.17，去掉完全关联矩阵的最后一行，得到图的关联矩阵如下，此时结点 v_5 是参考点。

$$
B=\begin{matrix} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix}\begin{matrix} a & b & c & d & e & f & g \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix} \end{matrix}
$$

为了不同的应用，还定义了其他一些矩阵，如回路矩阵、距离矩阵、割集矩阵、传输矩阵等。其中一些矩阵，将在后面结合有关内容再作介绍。

§2.4 图在计算机里的存贮

图的计算往往比较复杂，通常都要利用计算机，因此怎样把图的数据存贮到计算机里是一个首先要解决的问题，一般需要根据具体的图以及将作的运算来灵活选用存贮结构，下面介绍几种常用的图的存贮结构。

一、邻接矩阵

用邻接矩阵表示图，很容易判定两个结点之间是否有边相连，也容易求出各结点的次数。邻接矩阵可用二维数组来表示，如果是无向图，由于其对称性，仅需存入下三角（或上三角）矩阵。

邻接矩阵的主要缺点就是占用的空间较大，要占用 $0(n^2)$ 的空间（n为图的结点数）。而且用直接法将邻接矩阵初始化也将占用较多的时间，因而很难改进算法的时间复杂性与空间复杂性的量级。

一种有效的改进方法是用二进制位向量来表示一个邻接矩阵的行（或列），因为多数计算机都有“与”、“或”、“异或”和“非”等逻辑运算，有些还有特殊的“位操作”运算，所以可以把一个机器字看成不是一个“数”而是每一位是互相独立的向量，这样一个机器字是一个二进制的位向量，用来表示邻接矩阵的行（或列），若对其进行位操作，运算速度要快得多。但位向量的数据结构也有缺点，即需要的字长一般远大于用其他方法所需的字长，否则就要用两个以上的机器字联合表示，运算就不方便了。

二、邻接表

邻接表也是图的一种常用表示方法，在这种存贮结构中，对图的每一个结点建立一个链表。第 i 个链表中的结点是与 v_i 邻接的结点（如果是有向图则是从 v_i 引出弧的终止结点）。每个结点由两个域组成：结点域用以指示与结点 v_i 邻接的点的序号，链域用以指向下一条边。每一个链表设一个表头结点，这些表头结点本身以向量的形式存贮，以便随机访问任一结点的链表。

图 2.29 表示无向图，图 2.30 表示有向图，它们的邻接表各列于其右。

在无向图中，有 n 个结点 m 条边，则它的邻接表需 n 个头结点和 $2m$ 个表结点，每个表结点有两个域，显然在边稀疏的情况下，用邻接表比用邻接矩阵节省存贮单元。

在无向图的邻接表中，第 i 个邻接表中的结点数即为结点 v_i 的次数，对于有向图，则

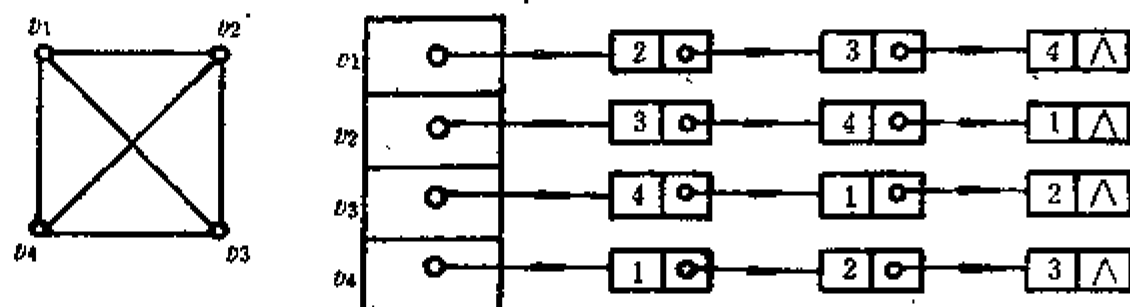


图 2.29　　　　表中的"∨"表示指针为空。

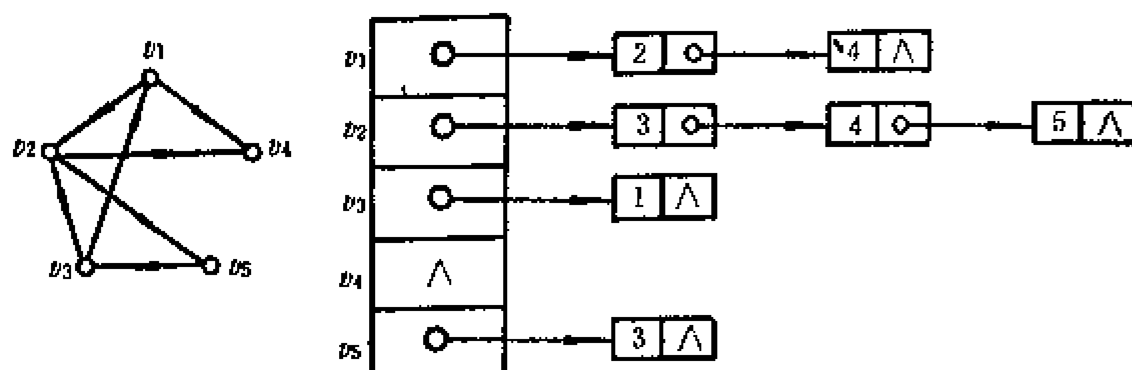


图 2.30

只是结点 v_i 的引出次数，如果要求它的引入次数，必须对整个邻接表扫描一遍，找到结点域的值为 v_i 的结点个数，显然这样做是比较麻烦的。为了确定各结点的引入次数，我们可以建立一个逆邻接表，即对每个结点 v_i 建立一个指向 v_i 的弧的表。图 2.31 给出了图 2.30的逆邻接表。

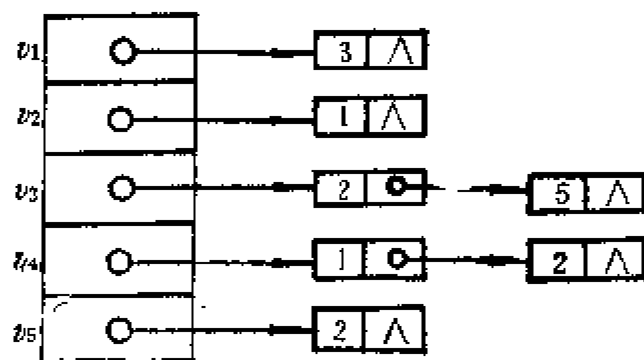


图 2.31

三、邻接向量

可以把邻接表改为对每个结点的邻接向量，用来表示结点与边的关系。结点 v_i 的邻接向量 V_i 是这样一个向量：它的各个分量是与结点 v_i 邻接的各结点的编号。例如图2.30的有向图的五个结点对应的邻接向量表示如下：

$$V_1=\{2,4,0\},\quad V_4=\{0,0,0\}$$
$$V_2=\{3,4,5\},\quad V_5=\{3,0,0\}$$
$$V_3=\{1,0,0\}$$

由于各结点的引出次数不同，它们的邻接向量的实际长度是不一样的。为了取得长度的统一，我们以引出次数最大的值作为向量的维数，对引出次数小的结点，就在向量的后段各分量置零。当然这样做要浪费一些空间。在对边的扫描次数不大的算法中，邻接向量是一种较好的表达方式，对于路径寻找算法和深度优先搜索法是很方便的。

四、邻接多重表

对于无向图，如果用邻接表的存贮结构，每一条边 (v_i, v_j) 有两个结点，一个在 V_i 的链表中，另一个在 V_j 的链表中，对一些较为复杂的问题，有时需要对搜索过的边作出记号，这样就必须同时找到表示同一条边的两个结点，如用邻接多重表作为图的存贮结构，就会方便得多。在邻接多重表中，每一条边用一个结点表示，它由如下所示的五个域

组成：

mark	V_i	V_j	V_i—link	V_j—link

其中，mark为标志域，用来标志该条边是否搜索过，V_i 与V_j 为该条边关联的两个结点，V_i—link 用来指向下一条关联 v_i 的边，V_j—link 则是指向下一条 关 联v_j 的边。在这种表中，除了增加一个标志域外，所需的存贮量与邻接表所需的相同。

例如，图 2.32 就是图 2.29 的邻接多重表。

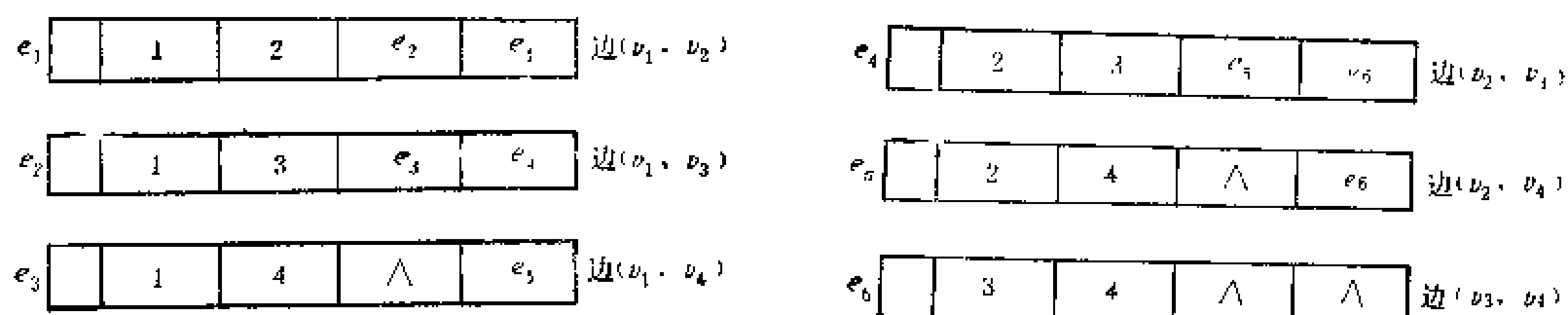


图 2.32

五、关联矩阵

用关联矩阵表示图，它的明显优点是能迅速指出与某一结点 v 关联的是哪些边，而且还可指出某条边 e 关联的是哪两个结点，对于有向图还能区分出结点的引入次数及引出次数。因为关联矩阵是 $n\times m$ 阶矩阵，其中大部分元素是零，空间的浪费是很大的。不过如果空间不紧张，这种存贮方式还是可取的，因为它可以获得时间高效的算法，空间的损失可以从时间上的获益得到补偿。

对于带权图（网络），存贮结构将复杂些，例如邻接表，每个结点还应增加一个 域 以存放相应的权值。

§2.5 图 的 遍 历

对于无向图，我们有时需要从图的某一点出发，访问图的所有其余结点，如果图是连通的，则从图的任一点出发顺着某些边可以访问图中的所有结点，而且使每一个结点仅被访问一次，这一过程称为图的遍历。

由于图的任一结点都可能与其他若干结点邻接，所以当访问了某个结点之后，可能顺着边又访问到已被访问过的结点，故在遍历图时，必须记下已被访问过的结点，以免同一结点被访问多次。因此我们可以对每个结点设置一个辅助函数 $N(v)$，当结点 v 未 被 访 问时，其值为零，一旦 v 被访问，便置 $N(v)$为 1 。

通常有两种遍历图的方法，即深度优先搜索法和广度优先搜索法，分别 记 作 DFS 算法和 BFS 算法，分述如下。

一、深度优先搜索法

算法的思路是：首先选定图 $G=(V,E)$ 中某一结点 v_0 为始点，设（v_0，u）是图中的一条边，于是沿着边（v_0，u）搜索到结点 u，设（u，w）是与 u 关联并且没有搜 索 过的

边，于是又沿着这条边搜索到结点 w。再从 w 出发重复上述过程。一般来说，设 x 是最新搜索（访问）到的结点，如果与 x 邻接的结点 y 尚未被访问过，则可沿边（x，y）访问到结点 y，如果与 x 邻接的所有结点都已被访问过，则退回到访问 x 的前一结点（称为 x 的先驱点），再重复上述过程，如此继续下去，直到所有的被访问过的结点的邻接点都已被访问为止。用这种搜索方法去遍历图，就称为深度优先搜索法。

如果图的存贮结构采用邻接表，则深先搜索的算法步骤如下：

设 v_0：起始点。

$P(v)$：v 的先驱点。

$N(v)$：v 的标号，如 v 已被访问，则 $N(v)=1$

T：输出的图。

输入无向图 G 的每个结点的邻接表 $L(v_i)$

1）$T\leftarrow\phi$，对所有的 $v\in V$，$N(v)\leftarrow 0$，$P(v)\leftarrow 0$，$P(v_0)\leftarrow v_0$，$v\leftarrow v_0$，$i\leftarrow 0$

2）$N(v)\leftarrow 1$

3）对 $L(v)$ 的每个结点 u，如都有 $N(u)=1$，则转5)，否则

4）$L(v)$ 中存在结点 u，$N(u)=0$，则 $T\leftarrow T\cup\{(v,\ u)\}$，$i\leftarrow i+1$，$P(u)\leftarrow v$，$v\leftarrow u$，转2)

5）如 $i=n-1$，则输出 T，停机。否则，$v\leftarrow P(v)$；转 3)。

例 2.18 用深度优先搜索法遍历图 2.33（a)。(b）是它的邻接表。则搜索过程如下

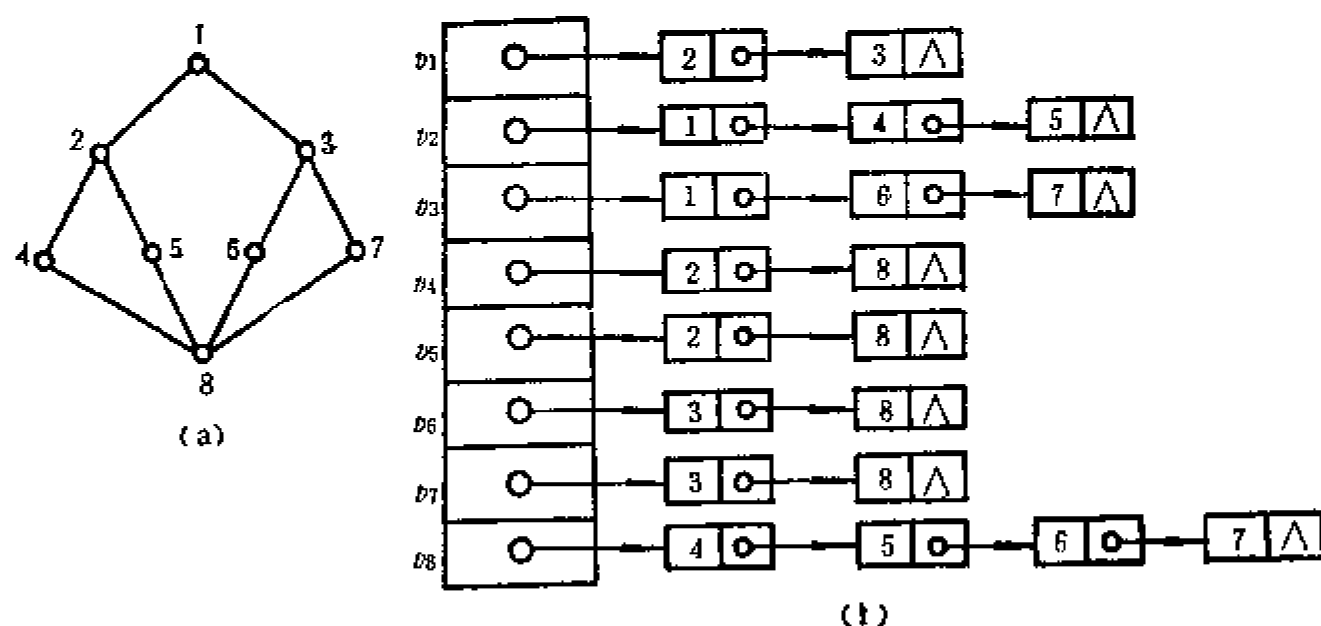


图 2.33

1）以结点 1 为始点，$N(1)=1$，找到未被访问过的邻接点 2，于是 $T=\{(1,2)\}$，$i=1$，$P(2)=1$，$v\leftarrow 2$，重复上述过程。

2）$N(2)=1$，找到与 2 邻接且未被访问过的点 4，于是 $T=\{(1,2),(2,4)\}$，$i=2$，$P(4)=2$，$v\leftarrow 4$，重复上述过程。

3）$N(4)=1$，找到与 4 邻接且未被访问过的点 8，于是 $T=\{(1,2),(2,4),(4,8)\}$，$i=3$，$P(8)=4$，$v\leftarrow 8$，重复上述过程。

4）$N(8)=1$，找到与 8 邻接且未被访问过的点 5，于是 $T=\{(1,2),(2,4),(4,8),(8,5)\}$，$i=4$，$P(5)=8$，$v\leftarrow 5$，重复上述过程。

5）$N(5)=1$，因与5邻接的点都已访问过，而且此时$i\neq n-1$，故$v\leftarrow P(5)=8$，重复前面过程。

6）找到与8邻接且未被访问过的点6，于是$T=\{(1,2),(2,4),(4,8),(8,5),(8,6)\}$，$i=5$，$P(6)=8$，$v\leftarrow 6$，重复上述过程。

7）$N(6)=1$，找到与6邻接且未被访问过的点3，于是$T=\{(1,2),(2,4),(4,8),(8,5),(8,6),(6,3)\}$，$i=6$，$P(3)=6$，$v\leftarrow 3$，重复上述过程。

8）$N(3)=1$，找到与3邻接且未被访问过的点7，于是$T=\{(1,2),(2,4),(4,8),(8,5),(8,6),(6,3),(3,7)\}$，$i=7$，$P(7)=3$，$v\leftarrow 7$，重复上述过程。

9）$N(7)=1$，因与7邻接的点都已访问过，而且此时$i=n-1=7$，故输出T，停机。

访问结点的次序为

$$v_1\rightarrow v_2\rightarrow v_4\rightarrow v_8\rightarrow v_5\rightarrow v_6\rightarrow v_3\rightarrow v_7$$

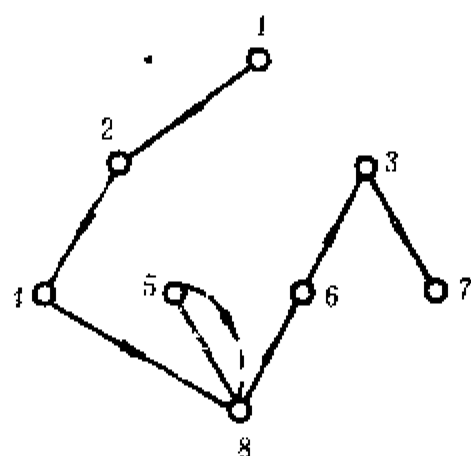


图 2.34

遍历的路线如图2.34。可以看出，访问的路线不仅与起始点的选定有关，而且与邻接表给定的顺序也有关。

采用邻接表这种存贮结构，由于对每条边至多运算一次，故时间复杂性为$O(|E|)$。如果用邻接矩阵表示图，时间复杂性将是$O(n^2)$。

上面的算法只适用于连通无向图，如果是非连通无向图或有向图，必须修改算法才能做到图的遍历。下面介绍一个算法，它是按递归过程写出的，既适于连通或非连通无向图，也适于有向图。

这一算法，图的输入仍采用邻接表，输出为边集F，对每一结点v设置一标记DFI(v)，它的初值为零，终值就是遍历图时访问结点v的次序号，称为结点v的深度优先指数。结点的这一序号清晰地显示出遍历图所走的路线。算法步骤如下，对于连通无向图，步骤11）可以删去，而把步骤12）的u作为起始结点。

```
1)          Procedure DFS(v)
                begin
2)                DFI (v)←i
3)                i←i+1
4)          for  L(v)中每个结点v' do
5)              if DFI(v')=0  then
                    begin
6)                  F←F∪{(v, v')}
7)                    DFS(v')
                    end
            end of DFS
8)                i←1
9)                F←φ
```

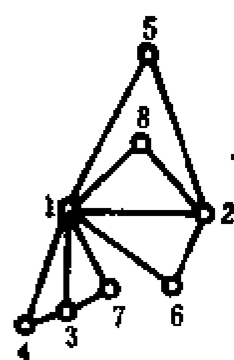
10)　　　　　for V 中每一个结点 v do DFI(v)← 0

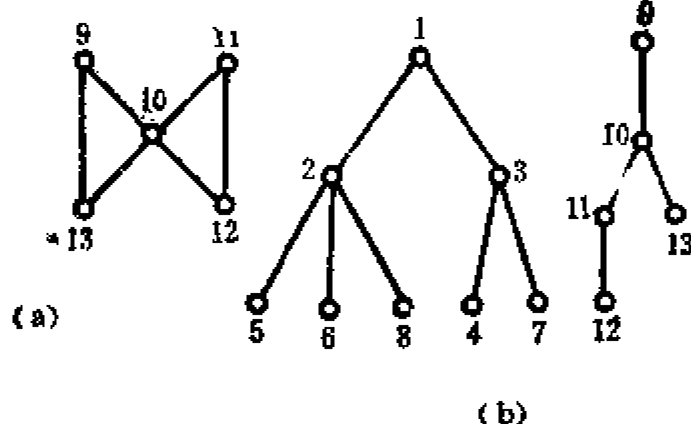
11)　　　　　while 存在某些结点 u，DFI (u) = 0　do

12)　　　　　　　DFS(u)

13)　　　　　　　output F

用上述算法遍历图2.35(a) 的非连通无向图，(b) 为它的输出，(c) 是遍历图时访问结点的次序号 $DFI(v)$。





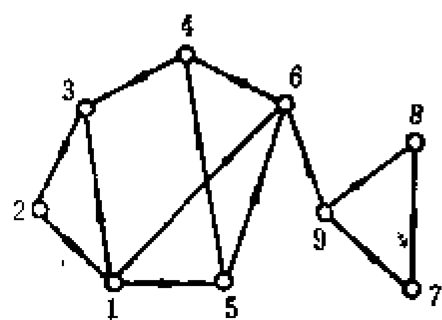
v	DFI(v)
1	1
2	2
3	6
4	7
5	3
6	4
7	8
8	5
9	9
10	10
11	11
12	12
13	13

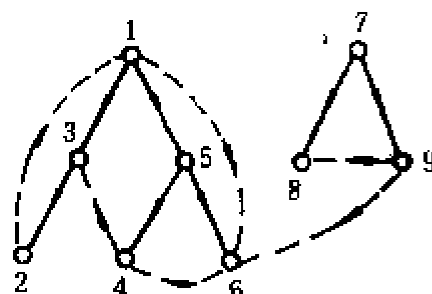
(c)

图 2.35

图 2.36(a)是一有向图，(b) 是遍历图输出，(c) 是遍历图时访问结点次序号。有向图的边可分为如下四种类型：

树 枝 集 $F=\{\langle 1,3\rangle,\langle 3,2\rangle,\langle 1,5\rangle,\langle 5,4\rangle,\langle 5,6\rangle,\langle 7,8\rangle,\langle 7,9\rangle\}$，后向弦集 $B_1=\{\langle 2,1\rangle\}$，前向弦集 $B_2=\{\langle 1,6\rangle\}$，跨接弦集 $C=\{\langle 4,3\rangle,\langle 6,4\rangle,\langle 9,8\rangle,\langle 9,6\rangle\}$。





v	DFI(v)
1	1
2	3
3	2
4	5
5	4
6	6
7	7
8	8
9	9

(c)

图 2.36

二、广度优先搜索法

算法的思路是：首先选定图 $G=(V, E)$ 的某一点 v_0 为始点，从 v_0 出发依次访问与 v_0 邻接的所有结点 $v_1, v_2, \cdots, v_t$，然后依次访问与 $v_1, v_2, \cdots, v_t$ 邻接的所有结点，（已被访问过的结点不再访问），依此类推，直到所有的结点都被访问过为止，按这种方法遍历图，就称为图的广度优先搜索法。

具体的算法取决于图的哪一种存贮结构，设仍采用邻接表的结构形式，广度优先搜索的算法步骤如下：

输入无向图 G 每个结点的邻接表 $L(v)$，建立一个先进先出的表（队列）Q，v_0 为始点。

（1）$T\leftarrow\phi$，对所有的点 $v\in V$，$N(v)\leftarrow 0$，$Q\leftarrow 0$，$i\leftarrow 0$，$v\leftarrow v_0$，$N(v)\leftarrow 1$。

（2）对 $L(v)$ 的每个结点 u，如都有 $N(u)=1$，则转(4)，否则

（3）$L(v)$ 中存在结点 u，$N(u)=0$，则 $T\leftarrow T\cup\{(v, u)\}$，$Q\leftarrow Q\cup\{u\}$，$i\leftarrow i+1$，$N(u)\leftarrow 1$，转(2)。

（4）如 $i=n-1$，停止，否则从 Q 中取排头元素 u，在 Q 中删除 u，$v\leftarrow u$，转(2)

例 2.19 如图2.33，采用广度优先搜索法遍历图，步骤如下

（1）以结点1为起始点，$N(1)=1$，找到未访问过的结点2，于是 $T=\{(1,2)\}$，$Q=\{2\}$，$i=1$，$N(2)=1$。

（2）再找与1邻接且未访问过的结点，得到结点3，于是 $T=\{(1,2),(1,3)\}$，$Q=\{2,3\}$，$i=2$，$N(3)=1$。

（3）因与1邻接的所有结点都已被访问过，而此时 $i=2\neq n-1$，故从 Q 中取出排头元素2，并从 Q 中删除2，即得 $Q=\{3\}$，$v\leftarrow 2$，执行步骤(2)

（4）找到与结点2邻接且未被访问过的结点4，于是 $T=\{1,2),(1,3),(2,4)\}$，$Q=\{3,4\}$，$i=3$，$N(4)=1$。

（5）再找与结点2邻接且未访问过的结点，得到结点5，于是 $T=\{1,2),(1,3),(2,4),(2,5)\}$，$Q=\{3,4,5\}$，$i=4$，$N(5)=1$。

（6）因与结点2邻接的结点都已访问过，而此时 $i\neq n-1$，故从 Q 中取出排头元素3且在 Q 中删除3，即得 $Q=\{4,5\}$，$v\leftarrow 3$，执行步骤(2)

（7）找到与结点3邻接且未被访问过的结点6，于是 $T=\{(1,2),(1,3),(2,4),(2,5),(3,6)\}$，$Q=\{4,5,6\}$，$i=5$，$N(6)=1$。

（8）再找与结点3邻接且未被访问过的结点，得到结点7，于是 $T=\{(1,2),(1,3),(2,4),(2,5),(3,6),(3,7)\}$，$Q=\{4,5,6,7\}$，$i=6$，$N(7)=1$。

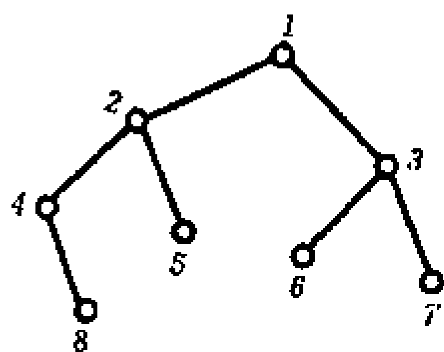


图 2.37

（9）因与结点3邻接的所有结点都已访问过，而此时 $i\neq n-1$，故从 Q 中取出排头元素4，并从 Q 中删除4，即得 $Q=\{5,6,7\}$，$v\leftarrow 4$，执行步骤(2)

（10）找到与结点4邻接且未被访问过的结点，得到结点8，于是 $T=\{(1,2),(1,3),(2,4),(2,5),(3,6),(3,7),(4,8)\}$，$Q=\{5,6,7,8\}$，$i=7$，$N(8)=1$。

（11）因与结点4邻接的所有结点都已访问过，且此时 $i=n-1$，故停止，输出 T。

访问点的次序为：

$$v_1 \to v_2 \to v_3 \to v_4 \to v_5 \to v_6 \to v_7 \to v_8$$

遍历图的路线如图2.37，同理，访问的路线不仅与起始点的选定有关，也与邻接表给定的顺序有关。容易看出，以上两种遍历图的算法的时间复杂性量级是等同的。这是因为两种算法的差别仅在于遍历图的过程中，结点访问的次序不同而已。

习题与思考题

1．证明：有向简单图所有结点引入次数的和，等于它的所有引出次数之和。并且这个和的数值等于边的数目。

2．证明：简单无向图结点的最大次数小于结点数。

3．证明：n个结点的简单无向图，至少有两个结点次数相同。（这里$n \geq 2$）

4．某次聚会的成员到会时握了手，试证：奇数个人握手的人数是一个偶数。

5．在一次象棋比赛中，任意两名选手间至多只下一盘，试证：总能找到两名选手，他们下过的盘数恰好相同。

6．在上题所指的比赛中，如果每名选手与其余所有的选手比赛过，且选手的人数是n，求比赛的总盘数。

7．证明：在n阶连通无向图中，如果有$n-1$条边，则至少有一个结点的次数为奇数。

8．画一个无向图，它的结点次数是

a）2，2，2，2，2，2

b）2，3，4，5，6

c）3，3，4，5，5，6

9．画出具有6个结点的图，使各结点的次数是1，2，2，3，5，5，这有的图有多少条边?

10．在n个运动队间安排了一项竞赛，已赛$n+1$局，试证：存在一个队，它至少参加过3局比赛。

11．试构造一个有$2n$个结点（$n \geq 3$）而没有三角形的三次正则图。

12．试证：具有n个结点的简单无向图，边的最大数目是$n(n-1)/2$。

13．在无向图中有一个结点集合，如果不在该集合中的每一个结点都邻接于该集合中一个或一个以上结点，则称该集合为支配集。如果支配集的任何真子集都不是支配集，则称该支配集为最小支配集。在图2.38中，试找出两个不同大小的支配集。

14．画出四阶完全无向图的所有生成子图。

15．画出图2.39(a)和(b)的补图。

16．证明：若无向图G是不连通的，则G的补图$\bar{G}$是连通的。

17．证明图2.40(a)(b)两个图是同构的，写出点和边的对应关系。

18．图2.41的(a)和(b)是否同构，如同构，写出它们的对应关系。

19．画出有三个结点的所有可能互不同构的的有向简单图。

20．求出具有5个结点，各结点次数都不小于3的简单无向图的个数。（画出图）

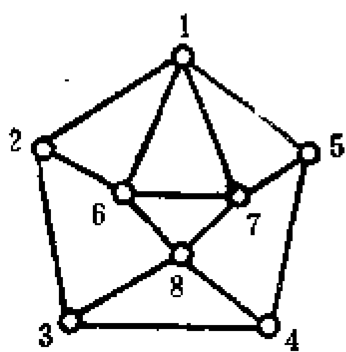


图 2.38

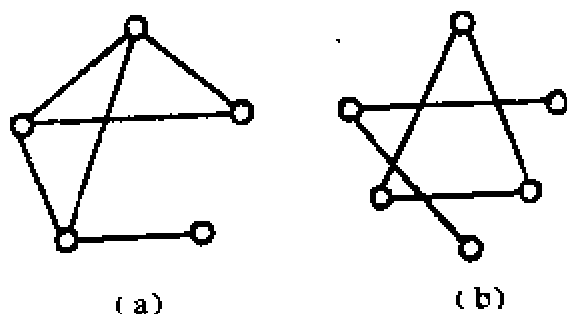


图 2.39

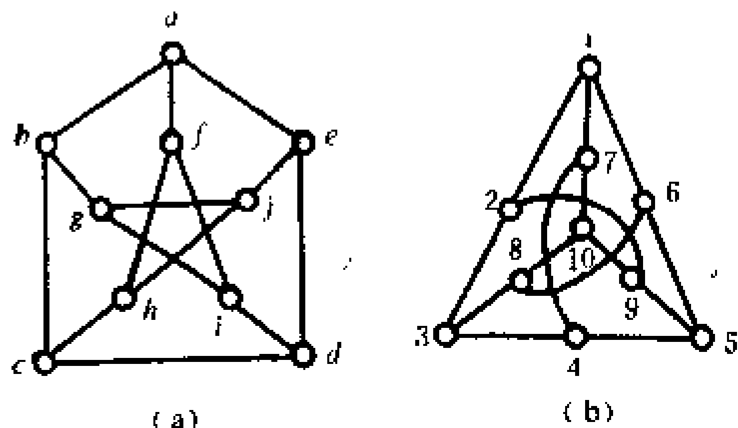


图 2.40

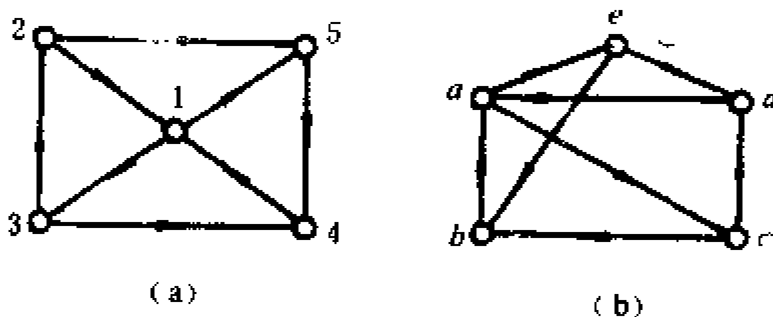


图 2.41

21．一个无向图如果同构于它的补图，则称该图为自补图。

（a）画出一个有 4 个结点的自补图。

（b）画出一个有 5 个结点的自补图。

（c）是否有 3 个结点、6 个结点的自补图。

22．证明：一个自补图或者有 $4K$ 个或者有（$4K+1$）个结点。（K为正整数）

23．证明：一个图是自补图，其对应的完全图的边数必为偶数。

24．图2.42给出了(a),(b),(c)三组 G_1，G_2图，试分别画出三组的G_1与G_2的交、并、差、补及环和的图。

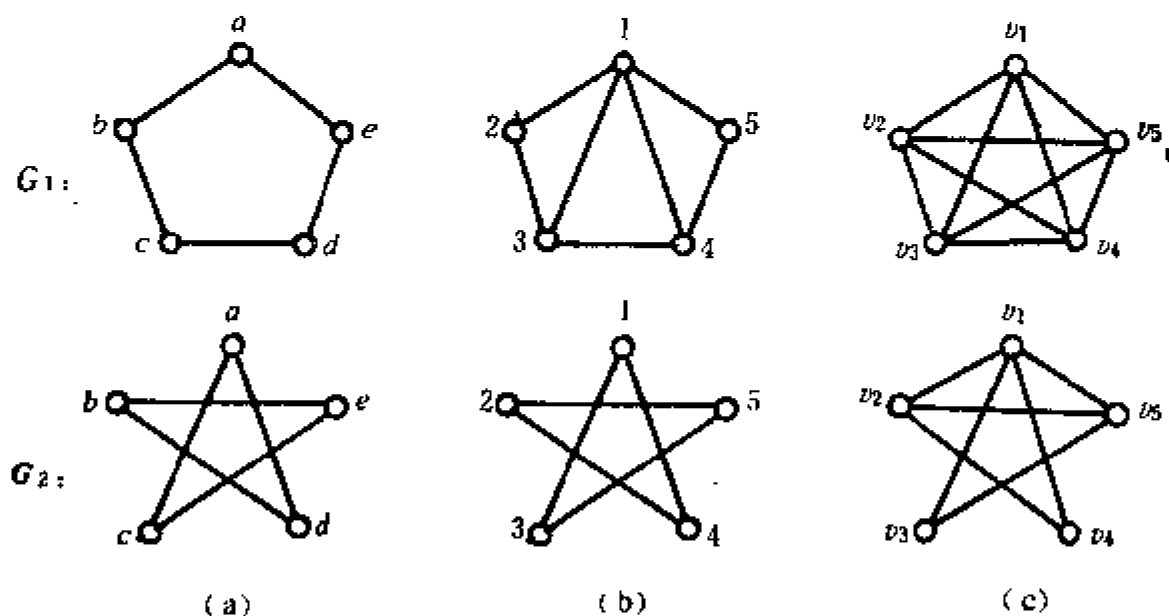


图 2.42

25．设 $G=(V,E)$是无向简单图但不是完全图，则一定存在三个结点 u，v，$w\in V$，有 $(u,v),(v,w)\in E$，但 $(u,w)\notin E$。

26．无向图 (n,m)，证明：如果$m\geqslant n$，则图必包含一条回路。

27．设G是一个 n 阶完全无向图

（a） G有多少回路

（b） 包含G的某条边 e 的回路有多少。

（c） 任意两点之间有多少条路径。

28．证明：一个有 n 个结点的简单图G，如果结点的最小次数 $deg(G)\geqslant 2$，则G包含一条长度不小于 $deg(G)+1$ 的回路。（$deg(G)=\min\{deg(v_i)|i=1,2,\cdots,n\}$）

29．在一个无向图G中，设 P_1 与 P_2 是两个给定点之间的两条不同路径。证明 $P_1\oplus P_2$ 是G中的一个回路或是回路的边不重并集。

30．确定图2.43的(a)，(b)各属于哪一连通类。

31．求出图2.43(b)中有向图的强分图。

32．图G有 n 个结点 m 条边，若

$$m>\frac{1}{2}(n-1)(n-2)$$

试证明G必是连通图。

33．证明：三次正则图有一个割点当且仅当它有一条割边。

34．证明：在一个有一条割边的三次正则图中至少有 10 个结点。

35．证明：若 v 是图G的一个割点，则 v 不是补图 $\bar{G}$ 的割点。

36．设G是一个至少有三个结点的连通图，下列命题是等价的。

（a） G没有割边。

（b） G的每二个结点在一条公共回路上。

（c） G的每一个点和一条边在一条公共回路上。

（d） G的每二条边在一条公共回路上。

（e） G的每一对结点和每一条边，有一条联结这两个结点而且通过这条边的路径。

（f） 对G的每一对结点和每一条边，有一条联结这两个结点而不含这条边的基本路径。

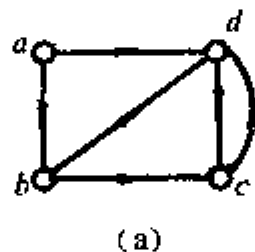
（g） 对每三个结点，有一条联结任何两个点而且含有第三个点的简单路径。

37．一个连通图，含有长为 m 的路径，不含任何长大于 m 的路径。求证：任两条长为 m 的路径有公共结点。

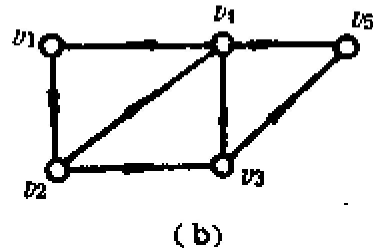
38．试证：一个简单连通无向图，若每个结点次数不小于 3，就含有长为偶数的一个回路。

39．求出图2.44中有向图的邻接矩阵 A，找出从 v_1 到 v_4 长度为 2 和 4 的路径，并用计算 A^2，A^3，A^4 来验证这一结论。

40．对于图2.44的有向图，试求出邻接矩阵 A 的转置 A^T 及 AA^T 和 A^TA，说明其中第



(a)



(b)

图 2.43

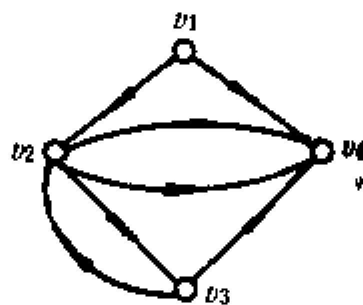


图 2.44

(2,3) 元素和第(2，2)元素的涵义。

41．给定两个简单有向图 G_1，G_2 的邻接矩阵 A_1，A_2 如下：

(a) 对于 $n=1, 2, \cdots, 6$，计算出 A_1^n 和 A_2^n。

(b) 求出 G_1 与 G_2 的所有基本回路

$$A_1=\begin{bmatrix}0&0&1&1&1&0\\0&0&0&0&1&1\\1&0&0&1&0&0\\1&0&1&0&1&0\\1&1&0&1&0&1\\0&1&0&0&1&0\end{bmatrix}\quad A_2=\begin{bmatrix}0&0&0&1&1&0&0\\0&0&0&0&0&1&1\\0&0&0&1&0&0&0\\1&0&1&0&1&0&0\\1&0&0&1&0&0&0\\0&1&0&0&0&0&0\\0&1&0&0&0&0&0\end{bmatrix}$$

42．无向图 G 的关联矩阵 B 如下，试画出 G 的图形并指出哪些是平行边。

$$B=\begin{array}{c}\\ v_1\\ v_2\\ v_3\\ v_4\end{array}\begin{array}{c}\begin{matrix}e_1&e_2&e_3&e_4&e_5&e_6\end{matrix}\\ \begin{bmatrix}0&1&1&0&1&0\\1&0&1&0&1&0\\0&1&0&1&0&1\\1&0&0&1&0&1\end{bmatrix}\end{array}$$

43．对于邻接矩阵 A 的简单有向图 G，它的距离矩阵定义如下：$D=(d_{ij})$，其中

$$d_{ij}=\begin{cases}k, & k \text{ 是使 } a_{ij}^{(k)}\neq 0 \text{ 的最小正整数}\\ 0, & i=j\\ \infty, & v_i \text{ 不能到达 } v_j\end{cases}$$

确定由图2.44所示图的距离矩阵，并指出 $d_{ij}=1$ 的涵意。

44．图2.45给出了一个有向图，试求该图的邻接阵矩及距离矩阵。

45．图2.46给出了一个简单有向图，试求出它的邻接矩阵及可达性矩阵。

46．在一简单有向图中，怎样才能从一个距离矩阵求得它的可达性矩阵。

47．设 $A=(a_{ij})_{n\times n}$ 是图 G 的邻接矩阵，试证：

(a) $(I+A)\wedge(I+A)=I+A+A^{(2)}$

(b) $(I+A)^{(n)}=I+A+A^{(2)}+\cdots+A^{(n)}$ 其中：$A^{(2)}=A\vee A$，"+"为布尔和，I 为 $n\times n$ 单位矩阵。

48．给定简单有向图 $G=(V,E)$，A 是 G 的邻接矩阵，P 是 G 的可达性矩阵，证明 $P=(I+A)^{(n)}$　"+"为布尔和。

49．试用 Warshall 算法求图 2.47 的可达性矩阵。

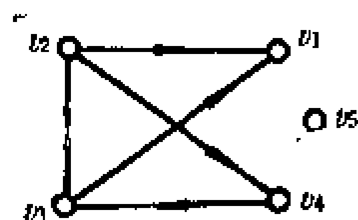


图 2.45

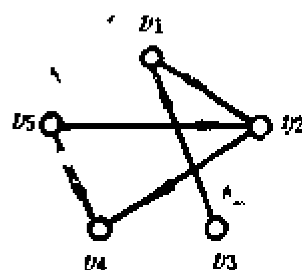


图 2.46

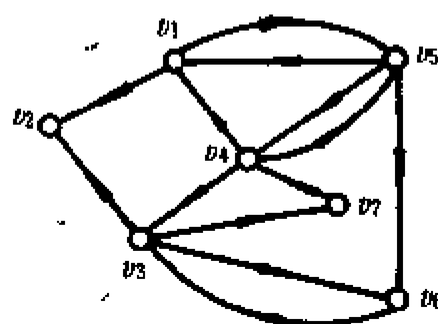


图 2.47

50. 试用可达性矩阵的运算，分析图 2.48 是否是强分图。

51. 试用一高级语言写出图的可达性矩阵的 Warshall 算法。

52. 试用一高级语言写出有向图的各强连通部分的算法。

53. 写出图 2.49 所示的图 G 的完全关联矩阵并验证其秩。

54. 设 $A=(a_{ij})_{n\times m}$ 是无向图 G 的完全关联矩阵，证明

$$\sum_{j=1}^{m} a_{ij}=d(v_i)$$

55. 试用深度优先搜索法遍历图 2.50 所示的无向图。

56. 试用广度优先搜索法重复上题。

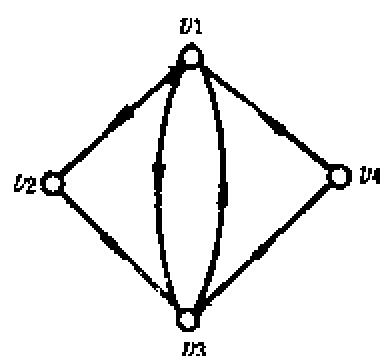


图 2.48

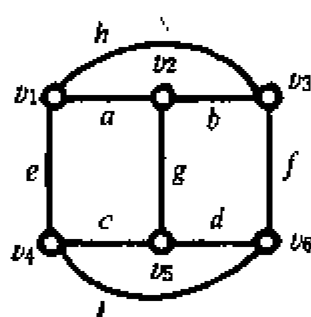


图 2.49

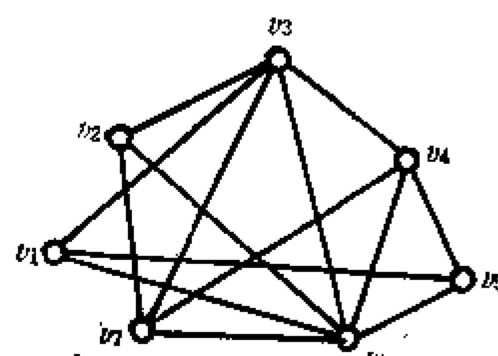
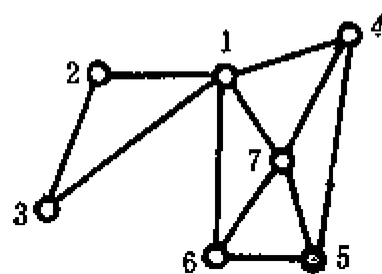
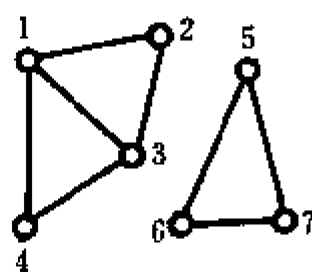


图 2.50

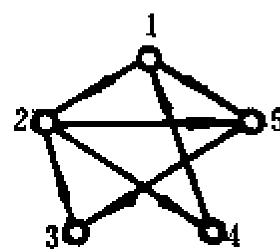
57. 试用深度优先搜索法的递归算法遍历图 2.51(a),(b),(c) 所示的连通无向图、非连通无向图及有向图。



(a)



(b)



(c)

图 2.51

58. 试用一高级语言写出连通无向图深先搜索程序。设图的输入用邻接矩阵。

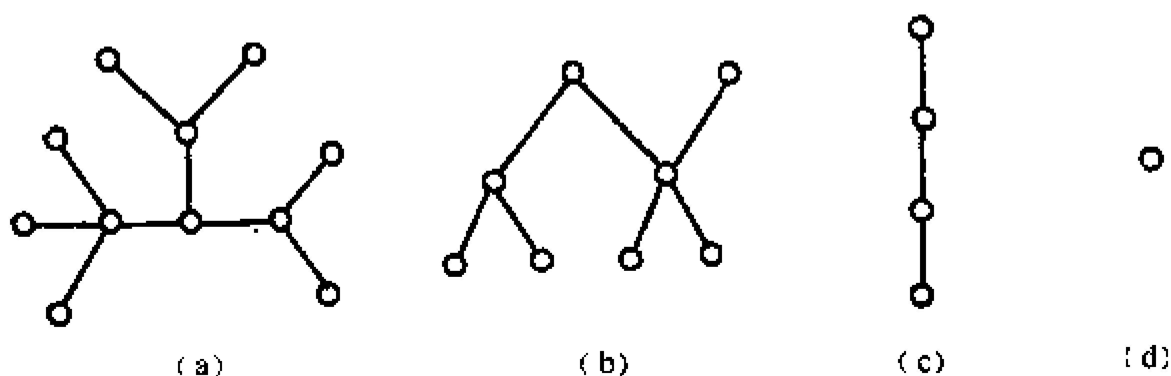
第三章 树与割集

§3.1 无 向 树

树是图论中最重要的一个概念，它在很多科学领域中得到广泛应用。下面我们将讨论树的基本特性、算法及其应用。

定义 3.1 一个连通且不含回路的无向图称为无向树，简称树。

图3.1中的（a)，(b)，(c)，(d）都表示无向树。



(a) (b) (c) (d)

图 3.1

常用符号 $T(n, m)$ 表示一棵树，其中 n 表示树的结点数，m 表示树的边数（树枝数)。次数为 1 的结点称为树的树叶，次数大于 1 的结点称为树的分枝点，边称为树枝。只有一个结点的树称为平凡树，它的次数为 0（无树枝)，如图中的（d）就是一棵平凡树。

定理 3.1 具有 n 个结点 m 条边的无向图 T 是树，当且仅当下列条件之一成立。

1）T 无回路且 $m=n-1$ (3.1)

2）T 连通且 $m=n-1$

3）任意两结点之间必然存在且仅存在一条基本路径。($n\geqslant 2$ 时)

4）无回路，如在任意两结点之间添上一条边，得到一个且仅一个基本回路，($n\geqslant 2$ 时)

5）图连通，但去掉任一条边图将不连通。($n\geqslant 2$ 时)

证：1）必要性：设 T 是树，根据定义 T 无回路。现对结点进行归纳证明 $m=n-1$。当 $n=1$ 时，$m=0$，公式成立。设 $n=k$ 时公式成立，即 $m=k-1$，现考察 $n=k+1$ 时的情况，即增加了一个结点。设为 v，下面证明结点 v 一定与原来 k 个结点中的一个结点连接，也只与 k 个结点中的一个结点连接。因为，如果 v 不与 k 个结点中的任一结点连接，则 v 是孤点，与树是连通的定义不符。如果 v 与 k 个结点中连接的点不止一个，不妨设两个点 v_1 与 v_2 都与 v 连接，即与 v 关联的边有两条 (v,v_1) 及 (v,v_2)，因为 v_1 与 v_2 是原来 k 个点中的任意两个点，它们原来是连通的，这样 v_1 与 v_2 将通过与 v 关联的这两条边形

成回路，与树无回路矛盾，因此，增加一个点一定增加也只能增加一条边，故此时边数为 $(k-1)+1=(k+1)-1=k$，公式 $m=n-1$ 亦成立。

充分性：设条件（1）成立，只要证明图连通，根据定义即得 T 是一棵树。采用反证法，设图不连通而是分成了 k 个连通分图 T_1，T_2，…，T_k，每一分图的结点数和边数用 n_i 和 m_i 表示（$1\leqslant i\leqslant k$），则 $n=\sum_{i=1}^{k} n_i$，$m=\sum_{i=1}^{k} m_i$，因每一个分图都是连通的而且无回路，所以每个分图都是一棵树，上面已证明应有 $m_i=n_i-1$，故 $m=\sum_{i=1}^{k} m_i=\sum_{i=1}^{k}(n_i-1)=n-k$，如果 $k\geqslant 2$ 则 $m=n-k<n-1$，与题设矛盾，故只能有 $k=1$ 才与题设相符，所以图是连通的，即图 T 是树。

2）必要性：设 T 是树，则 T 连通且由（1）已证 $m=n-1$，故条件（2）成立。

充分性：设条件（2）成立，只要证明图无回路即符合树的定义。现对 n 用归纳证明，当 $n=1$ 或 2 时，显然图无回路，命题成立。设 $n=k$ 时图无回路，这时边的数目为 $m'=k-1$，现考察 $n=k+1$ 时的情况，即增加了一个结点，设为 v，此时边的数目为 $m=k$，即比 m' 多了一条边，这条边的一个端点必然是 v，（否则 v 无关联边成为孤点，与图是连通的相矛盾）另一个端点是原来 k 个结点中的某一结点，由于原来 k 个点是无回路的，现在增加的这条边也不可能通过结点 v 构成回路，即 $n=k+1$ 时图也无回路，按定义图 T 是树。

3）必要性：设 T 是树，则图是连通的，任意两点之间必存在通路，如果这条通路不是基本路径，则通路中至少有一个结点的出现超过一次，因而必然含有回路。如果任意两点之间不止一条基本路径，也会形成回路，都与树的定义相矛盾，所以任意两点之间一定有一条也只能有一条基本路径，故条件（3）成立。

充分性：设条件（3）成立，则图是连通的，只要证明图无回路即符合树的定义，这是显然的，否则至少存在两个点它们之间不止一条基本路径，与前提矛盾。

4）必要性：设 T 是树，故 T 无回路，由（3）已证任意两点 v_i 与 v_j 之间有且只有一条基本路径，故添上一条边（v_i，v_j），只能得到唯一的一条基本回路，故条件（4）成立。

充分性：设条件（4）成立，因而图无回路，只要证明图是连通的，即符合树的定义。用反证法，如果任意两点之间不是连通的，则在这两点之间添上一条边就不可能得到一个回路。所以图必然是连通的，即图 T 是树。

5）必要性：设图 T 是树，则图是连通的，如果去掉任一条边（v_i，v_j）图仍连通，说明 v_i 与 v_j 之间还存在另一条通路，这条通路与边（v_i，v_j）形成回路，与树的定义矛盾。

充分性：设条件（5）成立，则图是连通的，如果去掉任意一条边图将不连图，说明图无回路，所以图 T 是树。 ▌

上面证明了无向树的定义与定理中的任一个条件都是等价的，从而也说明了定理中的任意两个条件都是彼此等价的。所以可以把其中的任一条作为无向树的定义，而将其余作为无向树的基本性质。这些性质对于我们分析研究无向树是非常重要的。

在上一章§2.2中，我们介绍了割点与割边的概念，现在把它们与树的性质联系起来，得到如下两条定理。

定理 3.2 当且仅当连通无向图的每一条边均为割边时，该图才是一棵树。

证： 必要性：设图G是一棵树，e是G的任一条边，因为树不含回路，所以e不在回路中，由定理2.10知，e是割边。

充分性：设G的任一条边均为割边，则去掉任一条边图将不连通，由定理3.1(5)知，图是树。 ▮

定理 3.3 无向树T的所有分枝点均为割点。

证： 设v是T的任一分枝点，则$deg(v)>1$，所以存在与v邻接的不同的结点u与w，因此uvw是T中u到w的唯一一条通路，若在T中去掉v及其关联边后，u与w将不通，按定义2.20，v是割点。 ▮

定理 3.4 任何一棵有两个以上结点的树至少有两片树叶。

证： 已知$m=n-1$，又$\sum\limits_{i=1}^{n}deg(v_i)=2m=2n-2$，因树是连通的，没有次数为0的结点（孤点），如果树没有一片树叶（即任一结点的次数都大于等于2），则$\sum\limits_{i=1}^{n}deg(v_i)\geqslant 2n$，与$\sum\limits_{i=1}^{n}deg(v_i)=2n-2$矛盾，如果树只有一片树叶，则$\sum\limits_{i=1}^{n}deg(v_i)\geqslant 2(n-1)+1=2n-1$，仍与上式矛盾，故至少有两片树叶。 ▮

定义 3.2 由不相连的树组成的图称为森林。

设F是t棵不相连的树组成的森林，n和m分别是F的结点总数和边的总数，则下式成立

$$m=n-t \tag{3.2}$$

§3.2 生 成 树

定义 3.3 若无向图G的生成子图T是一棵树，则称T为图G的生成树。

由定义可知，一个连通无向图可以有不止一棵生成树。如图3.2中(b)，(c)，(d)，(e)等都是图(a)的生成树。

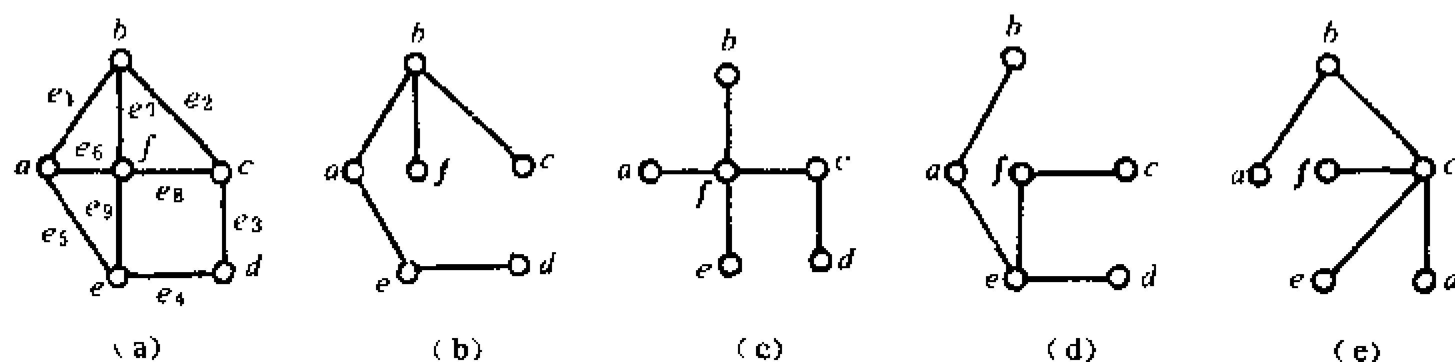


图 3.2

凡在图G中而不在生成树T中的边，称为对应于树T的弦，所有弦的集合称为树T的补。例如图3.2(c)的一棵生成树，e_3，e_6，e_7，e_8，e_9是它的树枝，e_1，e_2，e_4，e_5是它的弦，树T的补是$\{e_1, e_2, e_4, e_5\}$。弦的数目称为图G的圈数，记作$N(G)$，树枝的数目称为图的秩，记作$R(G)$。如果G有n个结点m条边，则$R(G)=n-1$，$N(G)=m-n+1$。

定理 3.5 一个连通图至少有一棵生成树。

证：如果连通图G无回路，则G本身就是一棵生成树。如果G有回路，去掉回路的任一条边得到生成子图G_1，显然G_1仍然是连通的，如果G_1不含回路，则G_1就是G的生成树，否则又可去掉回路的任一条边得到另一个生成子图，只要生成子图还有回路，就去掉回路的一条边，由于图的有限性，最后一定得到不含回路的生成子图T，由于每次去掉回路的一条边，并不破坏图的连通性，所以T是G的生成树。 ▮

定理的证明也为我们提供了一种构造生成树的方法，即每次去掉回路的任一条边，直到不再含有回路为止。这种方法称为破圈法。图 3.3 说明了它的构造步骤。

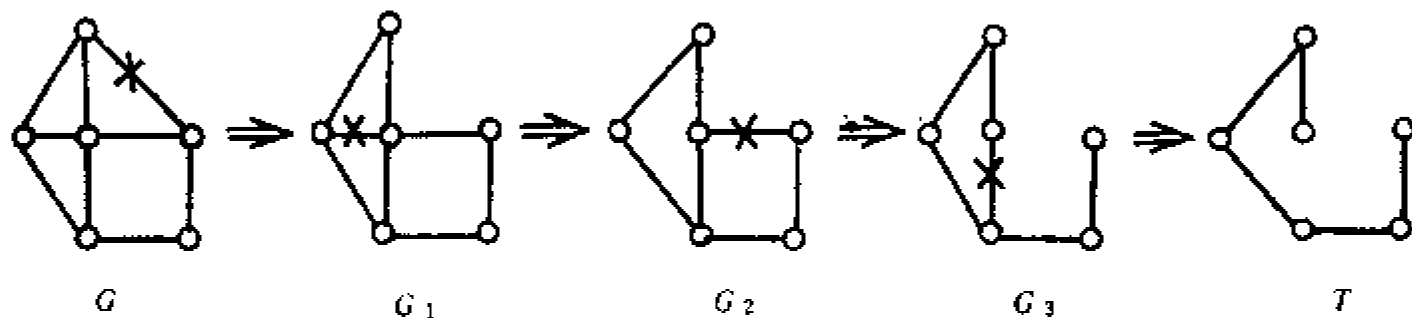


图 3.3

与破圈法相对应，还可采取一种称为避圈法的方法构造生成树，这种方法是：在G中任找一条边e_1，然后找一条不与e_1形成回路的边e_2，再找一条不与边集合$\{e_1, e_2\}$形成回路的边e_3，如此继续下去，使找的边都不与已找到的边集合形成回路，直到过程不能进行下去为止，则所有找到的边集合$\{e_1, e_2, \cdots, e_m\}$构成的图就是$G$的一棵生成树。图 3.4说明了它的步骤。

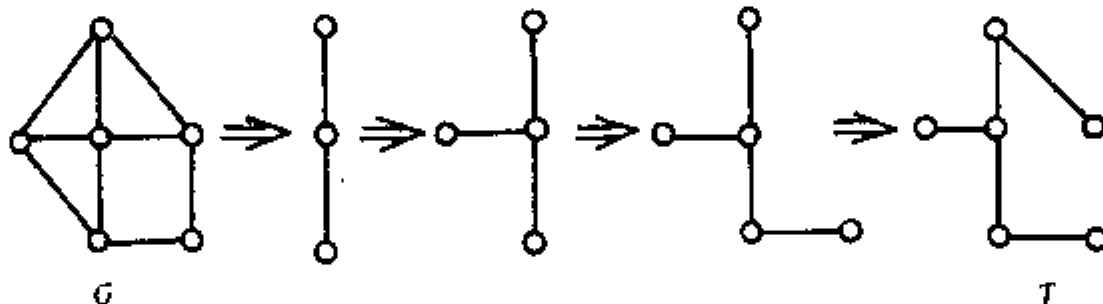


图 3.4

上述两种方法有一共同之处，就是每进行一步都要判定构造的图是否含有回路，所以这两种算法的计算量主要在回路的判定上。事实上，上一章我们讲图的遍历时，无论是深度优先搜索法还是广度优先搜索法，它们的输出都是图的生成树，所以都是构造图的生成树的算法，而且是比较好的两种算法。

定义 3.4 设T是图G的一棵生成树，由T的树枝和一条弦构成的回路称为对应于这条弦的基本回路，基本回路的集合，称为基本回路集。

如图 3.2(c)是(a)的一棵生成树，则

对应于弦e_1的基本回路是$C(e_1)=\{e_1, e_6, e_7\}$

对应于弦e_2的基本回路是$C(e_2)=\{e_2, e_7, e_8\}$

对应于弦e_4的基本回路是$C(e_4)=\{e_4, e_3, e_8, e_9\}$

对应于弦e_5的基本回路是$C(e_5)=\{e_5, e_6, e_9\}$。

树T的基本回路集是$C=\{C(e_1), C(e_2), C(e_4), C(e_5)\}$。

由于图的生成树不是唯一的，不同的生成树得到的基本回路也将不同，但基本回路的数目是相同的，就是图的圈数即弦的数目。

定理 3.6 设G是一连通无向图，则

1） G的任一回路至少含一条基本回路的弦。

2） 所有基本回路的环和不为空集，即

$$\bigoplus_{i=1}^{K} C_i \neq \phi, \quad k=N(G)=m-n+1$$

3） G的任一回路均可写成是若干个基本回路的环和。

证：1）如果有一回路不含任何基本回路的弦，即回路中的边全是树枝，它们一定包含在生成树中，但生成树是没有回路的，所以回路中至少有一条边不是树枝而是弦。

2）因为每一基本回路只有一条弦，不同的基本回路含有不同的弦，根据环和的性质，所有基本回路的环和不可能是空集。

3）设T是G的一棵生成树，B是G的任一回路，不失一般性，B包含边的集合写成

$$B=\{e_{i_1}, e_{i_2}, \cdots, e_{i_j}, e_{i_{j+1}}, \cdots, e_{i_m}\}$$

其中e_{i_k}是弦（$k=1, 2, \cdots, j$），e_{i_r}是树枝（$r=j+1, \cdots, m$），则含有上面这些弦的基本回路可写成$C_{i_1}, C_{i_2}, \cdots, C_{i_j}$，

设 $$B'=C_{i_1}\oplus C_{i_2}\oplus\cdots\oplus C_{i_j}$$

因此B'只含弦$e_{i_1}, e_{i_2}, \cdots, e_{i_j}$，不再含其他的弦，而$B$也只含这些弦，所以$B\oplus B'$将不再含有弦而只含有树枝，但是我们知道回路的环和仍是回路或是回路的边不重并集，因此只含树枝而不含弦是不可能的，这两种情况都要满足，只能有$B\oplus B'=\phi$成立，即$B=B'=C_{i_1}\oplus C_{i_2}\oplus\cdots\oplus C_{i_j}$。 ∎

所以任一回路B均可写成若干基本回路的环和，而且可以看出它们都是回路所含的弦所构成的基本回路。

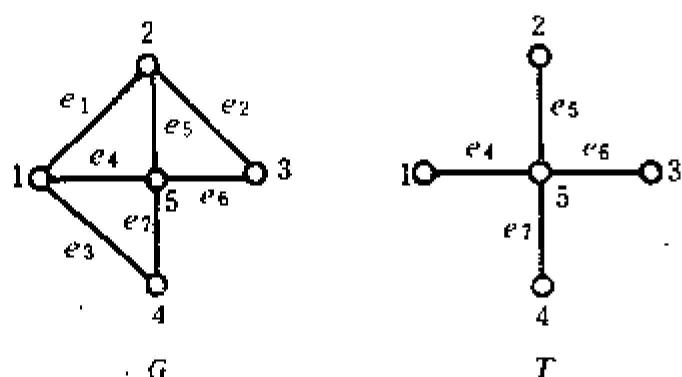


图 3.5

例 3.1 图3.5中T是G的一棵生成树，在G中任找一回路$B=\{e_1, e_2, e_4, e_6\}$，其中e_4，e_6为树枝，e_1，e_2为弦，对应e_1的基本回路$C_1=\{e_1, e_4, e_5\}$，对应e_2的基本回路$C_2=\{e_2, e_5, e_6\}$，故

$$B=C_1\oplus C_2=\{e_1, e_4, e_5\}\oplus\{e_2, e_5, e_6\}=\{e_1, e_2, e_4, e_6\}$$

下面讨论带权图的生成树

定义 3.5 设G是一个边带正数权的边权无向图，则称G的生成树为带权生成树，并以树枝所带权之和为生成树T的权，记作$C(T)$。

图3.6(a)为一边权无向图，(b)，(c)表示它的两棵带权生成树，其中$C(T_1)=92$，$C(T_2)=82$。

可见，生成树不同，树的权也不一样，其中一定存在权最大的生成树和权最小的生成树，它们是所有带权生成树中具有重要意义的生成树，下面我们着重讨论它们的求法，为此先给出定义如下。

定义 3.6 在带权图G的所有生成树中，树权最小的生成树称为图G的最小生成树。

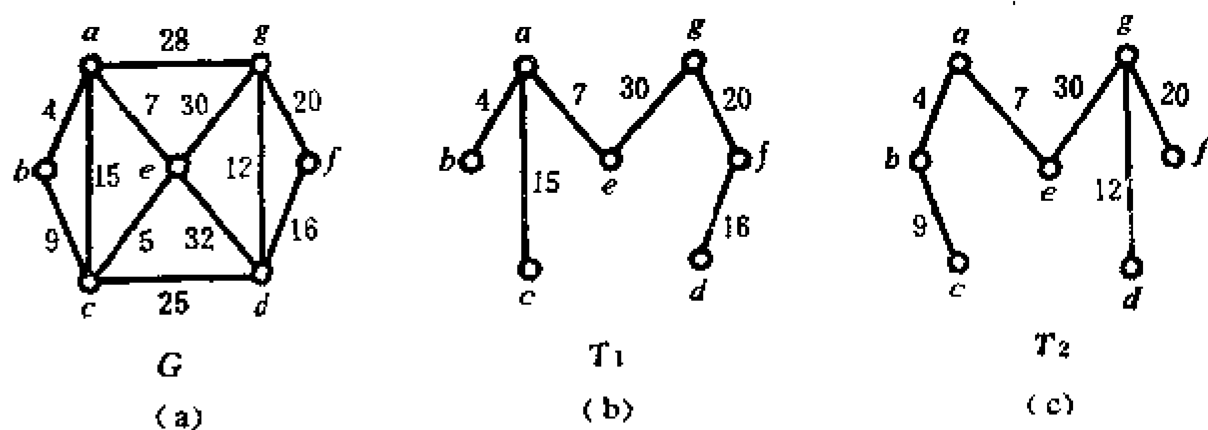


图 3.6

树权最大的生成树称为图G的最大生成树。

理论上讲，可以采用穷举法求一个带权图的最小（大）生成树，即把图的所有生成树都求出来，然后从中选取权最小（大）的，即是最小（大）生成树。实际上，当图的结点较多时，这种方法是行不通的。根据凯莱定理（证明从略），一个n阶完全图生成树的数目是n^{n-2}个，即使$n=30$，30^{28}也是一个40位的数，它是一个天文数字，就是使用当代最快速的计算机，也不是几代人所能够完成的。因此寻求简便而有效的算法，就是一项非常有意义的工作。下面以求最小生成树，介绍两种算法，它们在实际中都得到广泛的应用。

最小生成树算法一（Kruskal算法）

算法说明：输入无向图$G=(V, E)$及E中各边的权，输出为最小生成树T_0及它的权$C(T_0)$。算法使用了一个一维数组T_0，它的初始状态为空，算法结束时T_0中包含最小生成树的所有边。同理$C(T_0)$初始为空，结束时即表示最小生成树的权。VS是一个不相交的结点集合，初始状态时，$VS=\{\{v_1\}, \{v_2\}, \cdots, \{v_n\}\}$，算法的主要步骤是每次从边集$E$中选出一条未经处理的有最小权的边$(u, v)$进行分析，如果$u$和$v$同属于$VS$的一个元素集，则将$(u, v)$删去，如果$u$，$v$分属于$VS$的两个元素集，则将边$(u, v)$加到$T_0$中，并将这两个元素集合并为一个元素集，然后再从$E$中另选权最小的边进行处理，直到找出一棵最小生成树为止。

算法步骤：

1）$T_0 \leftarrow \phi$，$C(T_0) \leftarrow 0$，$VS \leftarrow \phi$，将E中的边按权从小到大排成序列Q

2）对所有$v \in V$，$VS \leftarrow \{v\}$，（这一步进行完毕后，$VS=\{\{v_1\}, \{v_2\}, \cdots, \{v_n\}\}$。）

3）如$|VS|=1$，输出T_0，$C(T_0)$，停机。否则进行下一步。

4）从Q中取出权最小的边(u, v)，并从Q中删除(u, v)。

5）如u，v在VS的元素集V_1，V_2中且$V_1=V_2$，则转（4），否则进行下一步。

6）$T_0 \leftarrow T_0 \cup \{(u, v)\}$，$V \leftarrow V_1 \cup V_2$，$C(T_0) \leftarrow C(T)+C(u, v)$，转（3）

算法正确性的证明：

按照算法得到的图记为T_0，由算法步骤可知，T_0有$(n-1)$条边且无回路，根据定理3.1，T_0是G的一棵生成树。下面证明它是最小生成树。用反证法，假设T_0不是最小生成树而T是最小生成树，则有如下两种可能：

（1）T_0与T有公共边$e_1, e_2, \cdots, e_i$，$(0<i \leqslant n-1)$，

如$i=n-1$，则$T_0=T$，命题得证。

如 $i<n-1$，则有 $e_{i+1}\in T_0$ 但 $e_{i+1}\notin T$，将 e_{i+1} 加到 T 中，必然形成回路 B，而 B 中至少有一条边不在 T_0 中（否则 T_0 含回路 B，与 T_0 是树矛盾），设这条边为 e'，将 e_{i+1} 加到 T 后并去掉 e' 将得到一棵新的生成树 T'，即

$$T'=(T-\{e'\})\cup\{e_{i+1}\}$$

则 T' 的权为 $$C(T')=C(T)-C(e')+C(e_{i+1})$$

已假设 T 是最小生成树，故 $C(T')\geqslant C(T)$，即 $C(e_{i+1})\geqslant C(e')$。已知 e_1，e_2，…，e_i，$e_{i+1}\in T_0$，而 e_1，e_2，…，e_i，$e'\in T$，说明 e_1，e_2，…，e_i 与 e' 不会形成回路，如果 $C(e_{i+1})>C(e')$，则在构造 T_0 的步骤中，选取 e_i 之后，将选取 e' 而不会选取 e_{i+1}，故只能有 $C(e_{i+1})=C(e')$，即 $C(T')=C(T)$，故 T' 也是最小生成树，但 T' 与 T_0 的公共边比 T 与 T_0 的公共边多了一条 e_{i+1}，（即有 $i+1$ 条公共边），如果 $i+1=n-1$，则 $T_0=T'$，所以 T_0 是最小生成树，否则用 T' 代替 T 再与 T_0 重复上述过程，直到两者的边完全相同为止，因而 T_0 是最小生成树。

（2）T_0 与 T 无公共边。可将 T_0 的第一条边 e_1 取代 T 的边 e' 得到一棵新的生成树 T'，前面已证明 T' 是最小生成树，且与 T_0 有了一条公共边，问题回到（1），即可按（1）得证。

根据上面的证明可以看出，当图的某些边具有相同的权时，最小生成树不是唯一的，但它们的权相同而且是最小的。

例 3.2 用 Kruskal 算法求图 3.6(a)的最小生成树。

解：将图的边按权从小到大排列如下：(a,b)，(c, e)，(a,e)，(b,c)，(d,g)，(a, c)，(d,f)，(f,g)，(c,d)，(a,g)，(e,g)，(d,e)，计算步骤列表如下。

步骤	选出边 e	$C(e)$	操作	VS	T_0	$C(T_0)$
1	(a,b)	4	加入 T 中	$\{\{a,b\},\{c\},\{d\},\{e\},\{f\},\{g\}\}$	$\{(a,b)\}$	4
2	(c,e)	5	同上	$\{\{a,b\},\{c,e\},\{d\},\{f\},\{g\}\}$	$\{(a,b),(c,e)\}$	9
3	(a,e)	7	同上	$\{\{a,b,c,e\},\{d\},\{f\},\{g\}\}$	$\{(a,b),(c,e),(a,e)\}$	16
4	(b,c)	9	删除	同上	同上	同上
5	(d,g)	12	加入 T 中	$\{\{a,b,c,e\},\{d,g\},\{f\}\}$	$\{(a,b),(c,e),(a,e),(d,g)\}$	28
6	(a,c)	15	删除	同上	同上	同上
7	(d,f)	16	加入 T 中	$\{\{a,b,c,e\},\{d,g,f\}\}$	$\{(a,b),(c,e),(a,e),(d,g),(d,f)\}$	44
8	(f,g)	20	删除	同上	同上	同上
9	(c,d)	25	加入 T 中	$\{\{a,b,c,d,e,f,g\}\}$	$\{(a,b),(c,e),(a,e),(d,g),(d,f),(c,d)\}$	69

表中第 9 步以后 $|VS|=1$，计算停止，输出 T_0 及 $C(T_0)$，T_0 的图形如图 3.7 所示。

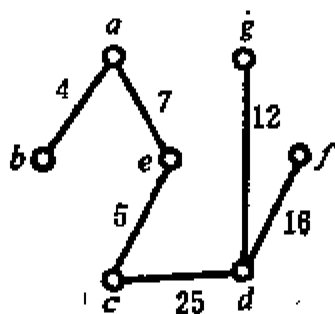


图 3.7

最小生成树算法二（Prim 算法）。

算法说明：先指定任意一点为初始访问点，记作 v_0，将 v_0 加到通过点集合 V' 中，然后找出跨接在通过点集合 V' 与未通过点集合 $V-V'$ 之间权最小的边 e 作为通过边加到最小生成树 T_0 的边集中去，并将 e 在 $V-V'$ 中的端点转到 V' 中。重复上述过程直到 $V'=V$ 为止，最后得到的 T_0 即是最小生成树的边集。

算法步骤：

1）$T_0 \leftarrow \phi$，$C(T_0) \leftarrow 0$，$V' \leftarrow \{v_0\}$

2）对每一点 $v \in (V-V')$，$L(v) \leftarrow C(v,v_0)$〔如 $(v,v_0) \notin E$，则 $C(v,v_0)=\infty$〕

3）如 $V'=V$，输出 T_0，$C(T_0)$，停机。否则进行下一步。

4）在 $V-V'$ 中找一点 u 使

$$L(u)=\min\{L(v) \mid v \in (V-V')\}$$

并记在 V' 中与 u 邻接的点为 w，$e=(w,u)$

5）$T_0 \leftarrow T_0 \cup \{e\}$，$C(T_0) \leftarrow C(T_0)+C(e)$，$V' \leftarrow V' \cup \{u\}$

6）对所有 $v \in V-V'$，如 $C(v,u)<L(v)$ 则 $L(v) \leftarrow C(v,u)$，否则 $L(v)$ 不变。

7）转（3）。

算法正确性的证明：

因为算法步骤是不断地从 V' 与 $(V-V')$ 之间找权最小的边 e 加到 T_0 中，现用反证法，设在 V' 与 $(V-V')$ 之间权最小的边 $e=(v_i, v_j)$ 不在 T_0 中，如图 3.8 所示，则将 e 加到 T_0 中必形成一回路，这个回路必然存在从 V' 到 $V-V'$ 的另一条边，设为 e'，则将 e 代替 e' 得到一棵新的生成树 T'，它的权将比 T_0 的权更小，与题矛盾。因此跨在 V' 与 $(V-V')$ 之间的最小边必是 T_0 上的边。

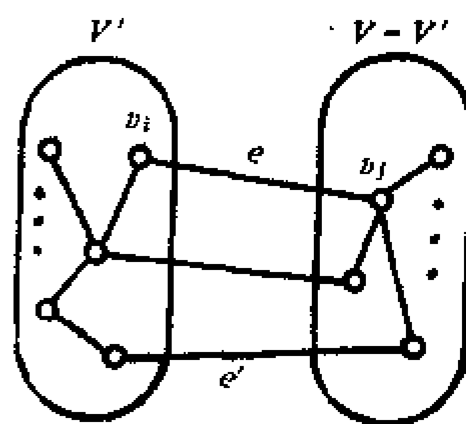


图 3.8

例 3.3 用 Prim 算法求图 3.6(a) 的最小生成树，计算步骤列表如下。

步骤	u	$L(b)$	$L(c)$	$L(d)$	$L(e)$	$L(f)$	$L(g)$	e	T'	T_0	$C(T_0)$
1	a	4	15	∞	7	∞	28		$\{a\}$	ϕ	0
2	b	—	9	∞	7	∞	28	(a,b)	$\{a,b\}$	$\{(a,b)\}$	4
3	e	—	5	32	—	∞	28	(a,e)	$\{a,b,e\}$	$\{(a,b),(a,e)\}$	11
4	c	—	—	25	—	∞	28	(c,e)	$\{a,b,e,c\}$	$\{(a,b),(a,e),(c,e)\}$	16
5	d	—	—	—	—	16	12	(c,d)	$\{a,b,e,c,d\}$	$\{(a,b),(a,e),(c,e),(c,d)\}$	41
6	g	—	—	—	—	16	—	(d,g)	$\{a,b,e,c,d,g\}$	$\{a,b),(a,e),(c,e),(c,d),(d,g)\}$	53
7	f	—	—	—	—	—	—	(d,f)	$\{a,b,e,c,d,g,f\}$	$\{(a,b),(a,e),(c,e),(c,d),(d,g),(d,f)\}$	69

可见计算结果与用 Kruskal 算法相同，输出最小生成树亦如图 3.7。

Kruskal 算法时间复杂性以 $O(m\log_2 m)$ 为界，当边数较多或是一个完全图时，$m \approx (n-1)^2$，则时间复杂性近似于 $O(n^2\log_2 n)$，而 Prim 算法的时间复杂性为 $O(n^2)$。所以如果图的连通度较高（最高为完全图），以 Prim 算法较好，如果图的连通度较低，当 $m \approx O(n)$ 时，则 Kruskal 算法可能更合适些。

最小生成树的理论和计算，在很多工程或技术领域中得到应用。例如要在若干城市之间架设通信线路、输电线路，铺设公路、铁路或各种管道，要求总的线路长度最短，或材料最省、成本最低等，这类问题归纳起来都是求最小生成树的问题。又如在印刷电路板上布线，不允许线路在非节点上相交，如何绘制连接线路才能使总的线路最短，也是一个求最小生成树的问题。

然而，如果图的边权代表的是利润或效益，就会成为求最大生成树的问题。从上面两

种算法可以看出，只要边权的选择改为从大到小，求最小生成树的算法，就可以用来求最大生成树了。

§3.3 有 向 树

有向树有内向树和外向树之分，外向树应用最广，下面仅讨论外向树。

定义 3.7 一个有向图，如果只有一个结点的引入次数为零，其余结点的引入次数均为1，则称为外向树（根树）。

在不讨论内向树时，常把外向树称为有向树而不加区别。

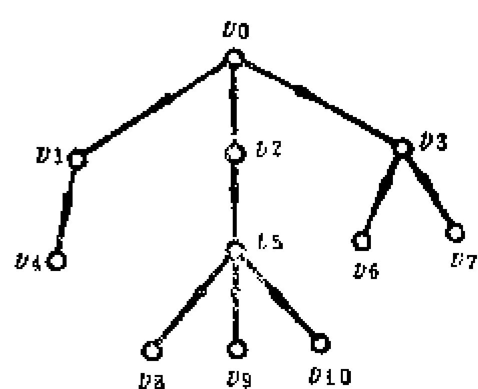


图 3.9

图 3.9 表示一棵有向树。其中引入次数为零的结点称为树的树根，引出次数为零的结点称为树叶，引出次数不为零的点称为分枝点。如图 v_0 是树根，v_4, v_6，v_7，v_8，v_9，v_{10} 是树叶，v_0，v_1，v_2，v_3，v_5 是分枝点。

通常，有向树的画法与生活中对树的画法恰好相反，即树根画在最上层而树叶画在最下层，因而边的方向都是从上向下的，在不致引起混淆时可以将边的箭头略去。

定义3.8 在有向树中，从树根到某一结点的通路中边的数目称为该点的长度(层次)，树叶的通路长度称为外部通路长度，分枝点的通路长度称为内部通路长度。结点通路长度最大的值，称为树的高度。

如图 3.9，树根 v_0 的层次为 0，v_1，v_2，v_3 的层次为 1，v_4，v_5，v_6，v_7 的层次为 2，v_8，v_9，v_{10} 的层次为 3。层次相同的结点都应画在同一水平上。

可以用一棵有向树表示家族关系，称为家族树，各结点之间的关系规定如下：

（1）从结点 v_i 经过一条边到达的结点，称为 v_i 的儿子，v_i 是他们的父亲。

（2）从结点 v_i 出发能到达的每一结点，称为 v_i 的后代，v_i 是他们的祖先。

（3）v_i 的所有儿子，他们互称为兄弟。

如图 3.9，树根 v_0 是所有点的祖先，v_2 则是v_5，v_8，v_9，v_{10} 的祖先，v_5 是 v_2 的儿子，也是 v_8，v_9，v_{10} 的父亲。v_8，v_9，v_{10} 是兄弟。以后我们讨论有向树时，常常引用家族树的这些称呼。

定义 3.9 如果对有向树的每一结点，都按次序给以编号，则称这样的树为有序树。

一般约定：在同一层上的结点，从左向右排序。也可用边的次序代替点的次序。

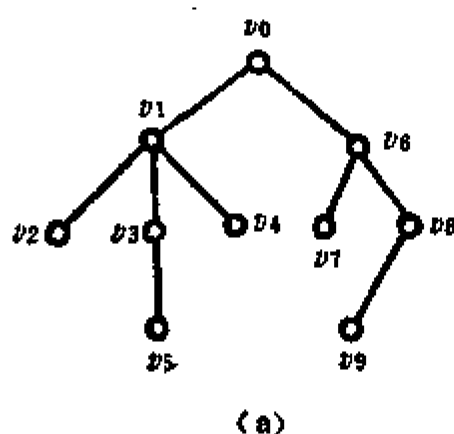
判定两棵有序树是否同构，不仅要求两棵树的点与点以及边与边有着一一对应的关系，而且它们的标号次序也必须相同。

定义 3.10 设 v 是有向树 $T=(V, E)$的一个分枝点，则以 v 为根的子树是 T 的一个子图 $T'=(V', E')$，其中 V' 包含结点 v 及它的所有后代，E'是从 v 出发的所有通路的边。

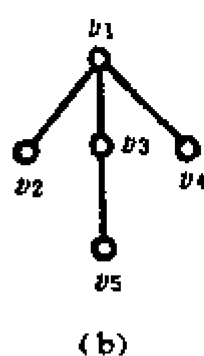
如图 3.10(a)是有向树 T，(b)是以结点 v_1 为根的子树 T_1，(c)是以结点v_6 为根的子树 T_2，(d)是以 v_2 为根的子树 T_3。

定义 3.11 有向树 T 的以结点 v 的一个儿子为根的子树，称为结点 v 的子树。

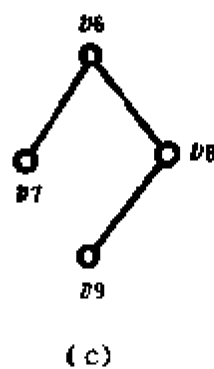
如图 3.10(a)的有向树 T，结点 v_1 的子树有三棵，如图 3.11(a)，(b)，(c) 所示。



(a)



(b)



(c)

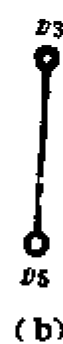


(d)

图 3.10



(a)



(b)



(c)

图 3.11

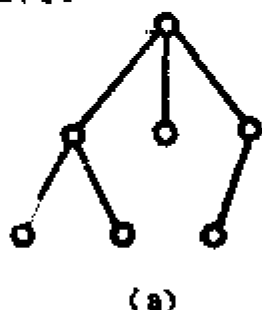
定义3.12 在有向树T中，

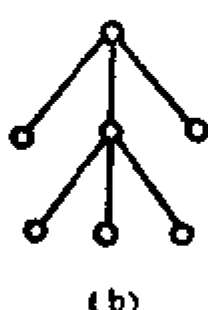
1）如果结点引出次数的最大值为m，则称T为m元树。

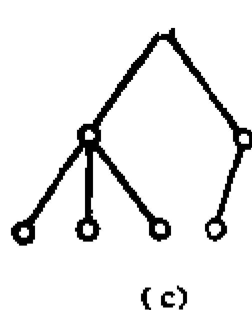
2）如果除树叶外每一结点的引出次数均为m，则称T为完全m元树。

3）如果树叶的层次均相同，则称T为正则树。

如图3.12，（a）是三元树，（b）是完全三元树，（c）是正则三元树，（d）是完全正则三元树。


(a)


(b)


(c)

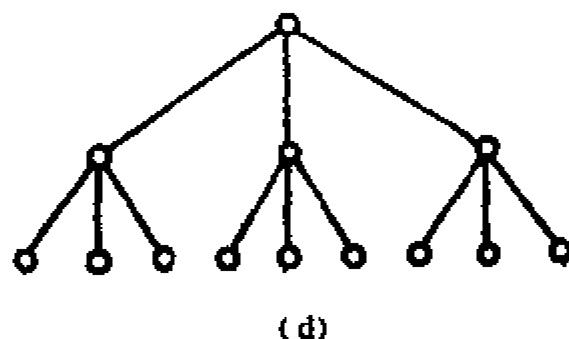

(d)

图 3.12

定理 3.7 设T为完全m元树，它的树叶数为t，分枝点数为i，则下式成立。

$$(m-1)i=t-1 \tag{3.3}$$

证：设树的结点数为n，边数为k，则有$k=n-1$，及$n=i+t$。因是完全m元树，除树叶外，每一结点都引出m条边，故边的总数$k=mi=n-1=i+t-1$，故$(m-1)i=t-1$。 ▮

最常用的m元树是二元树，因为它最简单，便于分析，也最容易用计算机进行处理，所以二元树是分析和研究多元树的基础。一棵m元树可以化成相应的二元树，步骤如下：

（1）对每一结点只保留它的左分枝，其余分枝都去掉。

（2）在同一层次上，原来是兄弟的结点，从左到右用有向边连接起来。（只从有引入边的结点连向无引入边的结点。）

（3）对任一结点按下法选定它的左儿子和右儿子，即在结点下面的结点是它的左儿子，右面连接的结点是它的右儿子。

（4）将结点的儿子画在下面一层，左儿子在左方，右儿子在右方，（即将图画成规范的二元树形式）

例 3.4 图3.13表示将一棵 4 元树化为相应二元树的过程。（d）就是（a）的相应二元树。

采用类似方法也可将一片森林化为一棵二元树。图 3.14 表示由两棵树组成的森林化

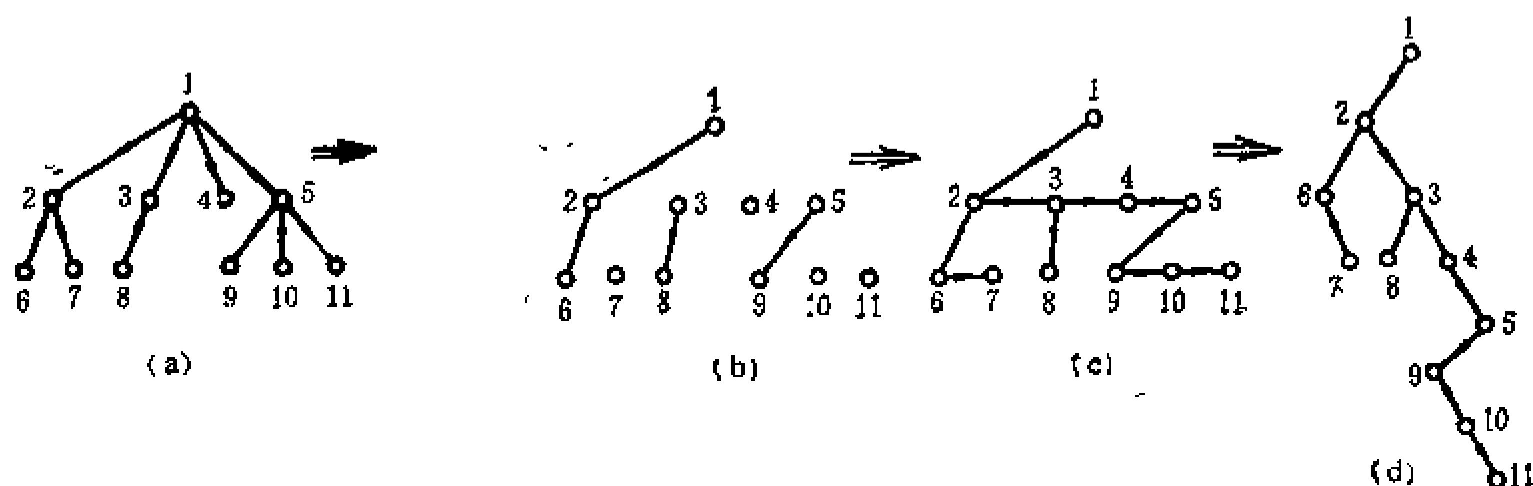


图 3.13

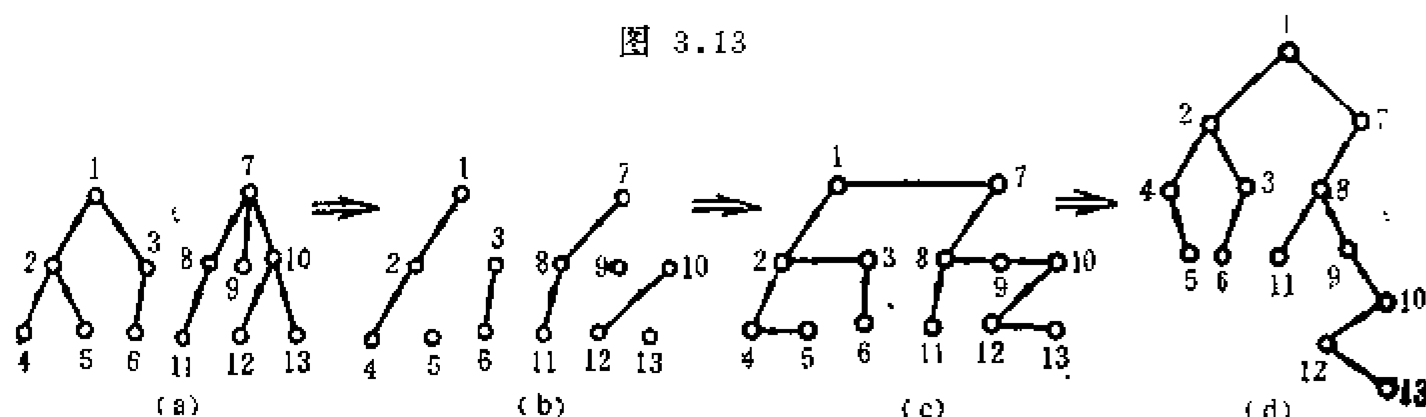


图 3.14

为二元树的过程。图中(a)为森林，(d)为相应的二元树。

画成规范形式的二元树时，左右儿子位置要分清楚。也可以将一棵二元树转化回相应的m元树，具体步骤请读者自己归纳。

定理 3.8 设T是一棵完全二元树，I为内部通路长度总和，E为外部通路长度总和，i为分枝点数，则下式成立

$$E = I + 2i \tag{3.4}$$

证：对分枝点用归纳法证明，当$i = 1$时，这时树有两片树叶和一个分枝点(树根)，如图3.15(a)，此时$I = 0$，$E = 2$，等式(3.4)成立。当$i = 2$时，树有两个分枝点(其中一个为树根)三片树叶，如图(b)，此时$I = 1$，$E = 5$，等式(3.4)亦成立。设$i = K$时等式成立，此时外部通路长度总和为E'，内部通路长度总和为I'，故有$E' = I' + 2K$。设$i = K + 1$时外部通路长度总和为E，内部通路长度总和为I。取树中儿子为树叶的任

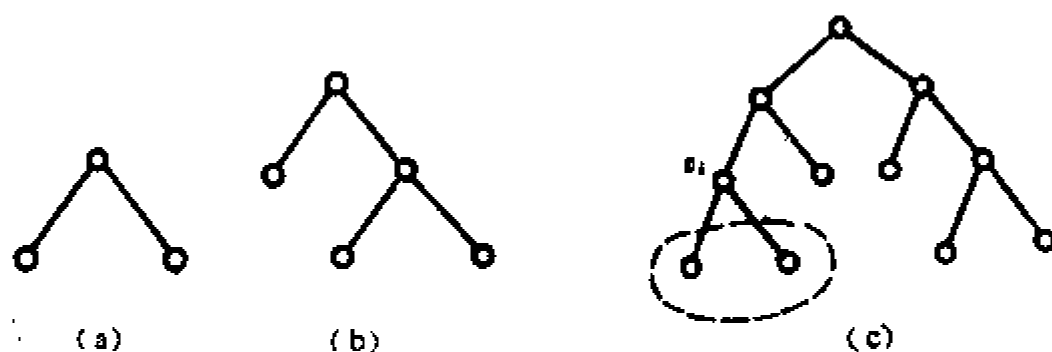


图 3.15

一分枝点v_i，如图(c)，设v_i的路径长度为I_i，现将它的子辈去掉得到一棵新的二元树T'，与原来的树比较，少了一个长度为I_i的分枝点，少了两片长度为(I_i+1)的树叶，多了一片长度为I_i的树叶，所以T'的内部通路长度总和为$(I-I_i)$，外部通路长度总和为

$(E-2(I_i+1)+I_i)$，但 T' 是有 K 个分枝点的二元树，根据假设前提，应有

$$E-2(I_i+1)+I_i=(I-I_i)+2K$$

故
$$E=I+2(K+1)，公式成立$$ ▮

§3.4 有向树的应用

树特别是二元树作为一种数据结构，在计算机科学技术中的应用是很广的，对此我们将在下两节予以讨论，这一节主要介绍其他方面的一些应用。

一、最优树

定义 3.13 一棵二元树 T 有 t 片树叶分别带权 w_1，w_2，…，w_t，并设 $w_1 \leqslant w_2 \leqslant \cdots \leqslant w_t$，则称此二元树为带权二元树。

定义 3.14 在带权二元树中，设带权为 w_i 的树叶 v_i 的通路长度为 $L(v_i)$，则称

$$W(T)=\sum_{i=1}^{t} w_i L(v_i) \tag{3.5}$$

为该带权二元树的权。

图 3.16 是有 4 片树叶的带权二元树，树叶所带的权分别为 12，5，6，7，树的权为 $W(T)=12\times1+5\times2+6\times3+7\times3=61$。

给定二元树各片树叶的权 w_1，w_2，…，w_t，我们可以构造出不同的带权二元树。例如给定了 4 片树叶分别带权 5，6，7，12，则除了可以构造出如图 3.16 的带权二元树之外，还可以构造出其他形状不同的二元树，如图 3.17（a），（b），（c）等都是。它们的树叶数目及所带的权都是相同的，但二元树的权却完全不同，有的权大有的权小，因此可以肯定在给定条件下，一定存在一个权最小的二元树，称为最优树，定义如下。

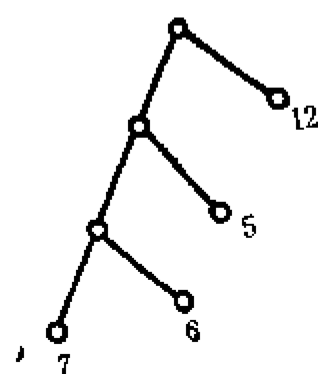


图 3.16

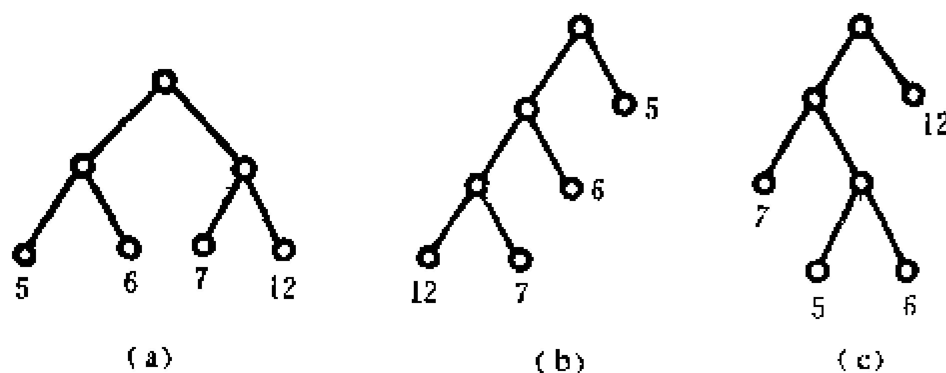


图 3.17

定义 3.15 在所有带权 w_1，w_2，…，w_t 的二元树中，$W(T)$ 最小的二元树称为最优树。

1952年，Huffman给出了最优树的算法，在介绍这一算法之前，我们先来分析最优树的特点。

定理 3.9 设 T 为树叶带权 $w_1 \leqslant w_2 \leqslant \cdots \leqslant w_t$ 的最优树，则带最小权的两片树叶是兄弟且其通路长度最长。

证：首先证明带最小权 w_1 的树叶的通路长度最长。用反证法，设带最小权 w_1 的树叶

v_1 的通路长度 $L(v_1)$ 不是最长的，则存在比 w_1 大的权 w_i 的树叶 v_i，它的通路长度 $L(v_i)$ 比 $L(v_1)$ 长，即 $w_i > w_1$，而 $L(v_i) > L(v_1)$，如图 3.18 所示。此时二元树的权可写成

$$W(T) = w_1 \cdot L(v_1) + w_i \cdot L(v_i) + Q$$

其中 Q 表示除 v_1 和 v_i 外其余树叶的权与其通路长度乘积之和。

现在将 v_1 与 v_i 上的权对换，其余树叶的位置不变，于是得到一棵新的二元树 T'，它的权应为

$$W(T') = w_1 \cdot L(v_i) + w_i \cdot L(v_1) + Q$$

则
$$W(T') - W(T) = (w_i - w_1)(L(v_1) - L(v_i)) < 0$$

即 $W(T') < W(T)$，则 T 不会是最优树，就是说只要通路长度最长的树叶不带最小权就不可能是最优树，故最优树中通路长度最长的树叶权必最小。

其次证明权最小的两片树叶必是兄弟。先证带最小权 w_1 的树叶 v_1 必有一片树叶作为它的兄弟，用反证法，设 v_1 没有兄弟，如图 3.19(a)，可以看出图(b)的权比(a)更小，

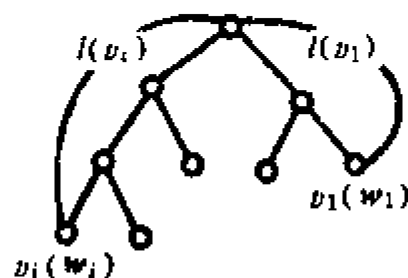


图 3.18

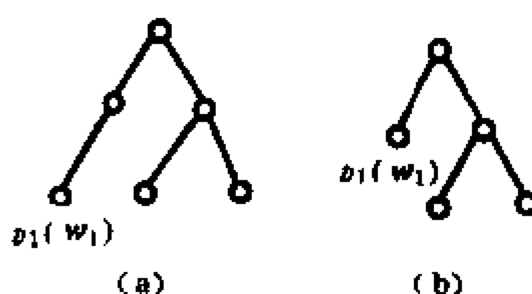


图 3.19

因此(a)不会是最优树。又设 v_1 虽有兄弟但不是树叶而是分枝点，如图 3.19(b)，则 v_1 的通路长度不是最长的，因而不可能是最优树，因此带最小权 w_1 的树叶 v_1 必有一片树叶作为它的兄弟。

其次证明作为 v_1 的兄弟的树叶所带的权是所有除 w_1 外最小的(即 w_2)，这与最小权的树叶通路长度最长的证明相同。

因此在最优树中，两片带最小权的树叶是兄弟，且通路长度最长。命题得证。

定理 3.10 设 T 是一棵带权 $w_1 \leqslant w_2 \leqslant \cdots \leqslant w_t$ 的最优树，去掉带最小权 w_1，w_2 的两片树叶，使它们的父亲为带权 $(w_1 + w_2)$ 的树叶，得到一棵新的带权二元树 T'，则 T' 也是最优树。

证：根据题设画出 T 与 T' 的示意图如图3.20(a)与(b)，在图(a)中，树叶 v_1，v_2 带最小权 w_1，w_2，它们的父亲为点 v，v 的通路长度为 $L(v)$，去掉 v_1，v_2 并使 v 带权 $(w_1 + w_2)$

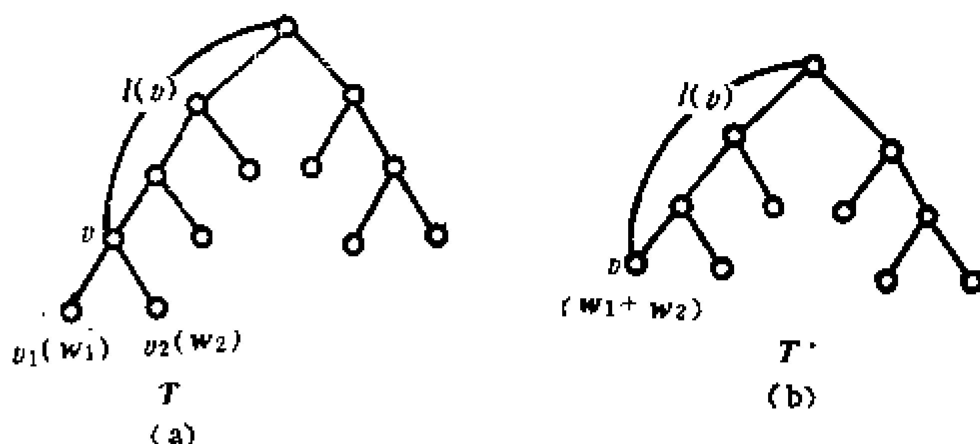


图 3.20

即得图(b)，因此可写出它们的权如下：

$$W(T)=(w_1+w_2)\cdot(L(v)+1)+Q$$
$$W(T')=(w_1+w_2)\cdot L(v)+Q$$

其中Q表示其余树叶的权与通路长度乘积之和。

则
$$W(T)=W(T')+w_1+w_2 \quad (1)$$

T'是有$(t-1)$片树叶分别带权(w_1+w_2)，w_3，…，w_t的树，如果它不是最优树，则设带这$t-1$个权的最优树为$\hat{T}'$，现将$\hat{T}'$中带权(w_1+w_2)的那片树叶作为分枝点，使它分出的两片树叶分别带权w_1和w_2，于是又得到一棵有t片树叶分别带权w_1，w_2，w_3，…，w_t的新的树，设为$\hat{T}$，显然$\hat{T}$与$\hat{T}'$的权有如下关系

$$W(\hat{T})=W(\hat{T}')+w_1+w_2 \quad (2)$$

由（1）－（2）得

$$W(T)-W(\hat{T})=W(T')-W(\hat{T}') \quad (3)$$

因T是最优树，应有

$$W(T)\leqslant W(\hat{T})\text{，即 }W(T)-W(\hat{T})\leqslant 0 \quad (4)$$

因$\hat{T}'$是最优树，应有

$$W(\hat{T}')\leqslant W(T')\text{，即 }W(T')-W(\hat{T}')\geqslant 0 \quad (5)$$

要同时满足(3)，(4)，(5)式只能有

$$W(\hat{T}')=W(T')\text{及 }W(T)=W(\hat{T})$$

即T'与$\hat{T}$都是最优树。 ▮

上面两个定理及其证明，实质上为我们提供了构造最优树的方法，就是 Huffman 提出的算法，故称为 Huffman 算法，构造出的最优树也称 Huffman 树。

Huffman 算法步骤如下：

（1）权最小的两片树叶是兄弟，去掉这两片树叶，并以它们的父亲作树叶，使它带的权是它的两个儿子带权之和，于是得到比前少一片树叶的树。

（2）对得到的树重复（1），最后得到树根。

（3）从树叶到树根的构造过程中得到的树，就是所求的最优树。

例 3.5 构造一棵最优树，它们树叶带的权分别为1，3，3，3，5，5，20，60。

解： 图3.21(a)～(g)表示了最优树的构造过程，(h)就是最优树。

二元树是一种很重要的非线性数据结构，在计算机中不仅应用很广，而且还常常需要考虑二元树的通路长度问题，即最优树的问题，此外，最优树在编码通讯中也起着重要的作用。

我们知道，在远距离通讯中，常用0和1的字符串表示英文字母进行信息的传送，英文有26个字母，所以只要用长度为5的0，1字符串就可以表示26个字母而不致混淆，发送时只要发送一条0和1的字符长串，它正好包含信息中字母对应的字符序列，在接收端，只要将这一长串字符按长度为5的序列分割，就可得到对应的字母，这种方法称为等长编码通讯法。处理等长的编码，对接收端来说是较容易的。

但是，由于字母在信息中出现的频繁程度不一样，例如字母e和t在单词中出现的次数要比q和z频繁得多，因此人们希望能用较短的字符串表示频繁出现的字母，这样就可

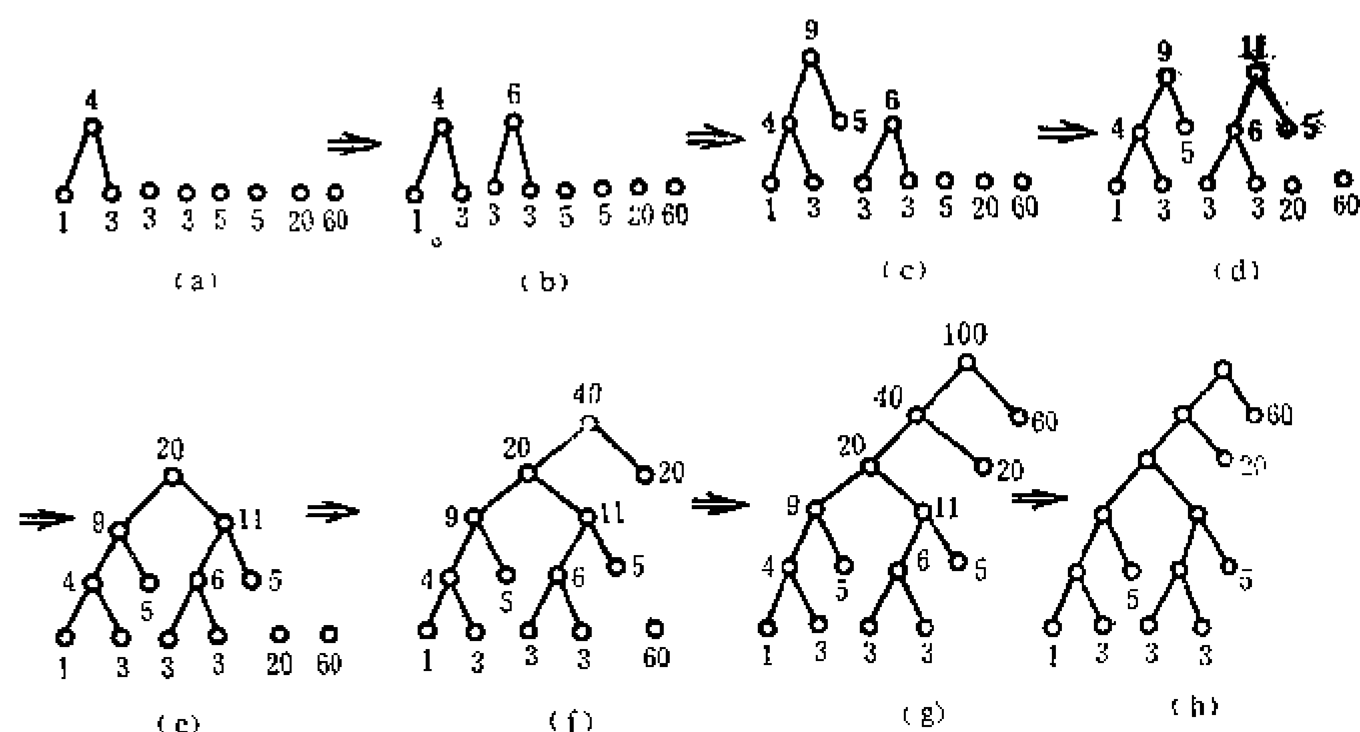


图 3.21

缩短信息字符串的总长度，显然如能实现这一想法是很有价值的。对发射端来说，传送长短不同的字符序列并无困难，但是在接收端，怎样才能明确无误地将收到的一长串字符分割成长度不同的序列呢？也就是说接收端如何进行译码呢？例如我们用 0 0 表示字母 t，用 0 1 表示 e，用 0001 表示 W，那么当接收到信息为0001时，如何判别信息是 t、e 还是w呢？为了解决这一问题，先引出前缀码的概念。

定义 3.16 由一个序列的第一个符号到序列中间的某个符号所组成的子序列，称为这个序列的前缀。

例 3.6 子序列 010，01，0100 都是序列 010011 的前缀，但 10，00，001，011 都不是它的前缀。

定义 3.17 如果在一个序列的集合中，没有任何一个序列是另一个序列的前缀，则称这个序列集合为前缀码。如果前缀码的字符只用 0 和 1，则称为二元前缀码。

例 3.7 集合{000，001，01，10，11}是一个二元前缀码，但集合{000，0001，00，01，11}则不是二元前缀码。

定理 3.11 一棵二元树的树叶对应一个二元前缀码。

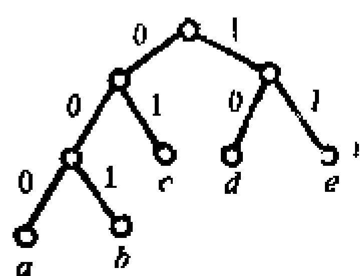


图 3.22

证：如图 3.22 表示一棵二元树，从分枝点向左引出的边标"0"，向右引出的边标"1"，每片树叶用从树根到它的通路上各条边的标号组成的 0，1 字符串表示，例如图中的 5 片树叶可表示为

a：000，　　b：001，　　c：01，

d：10，　　e：11

显然，没有任何一片树叶的标号是另一片树叶标号的前缀，因此一棵二元树的树叶标号对应一个前缀码。

定理 3.12 任何一个二元前缀码都对应一棵二元树。

证：设所给二元前缀码中最长序列的长度为 h，据此我们构造一棵高度为 h 的完全正则二元树，并按前面讲的方法给每条边标上"0"或"1"，并用从树根到某一结点通路

上各边标号形成的序列作为该结点的标号，因此二元前缀码的每一字符码都恰好与二元树中的一个结点的标号对应，以这些对应结点作为树叶（即将它们的后代及边都去掉）得的二元树，它的树叶标号就与二元前缀码对应。▮

例 3.8 求对应前缀码{000，001，01，1}的二元树。

解：前缀码中序列最长的长度为 3，因而构造一棵高度为 3 的完全正则二元树如图 3.23(a)，按照上述方法去掉冗余的结点和树枝，即可得到对应的二元树如图(b)。

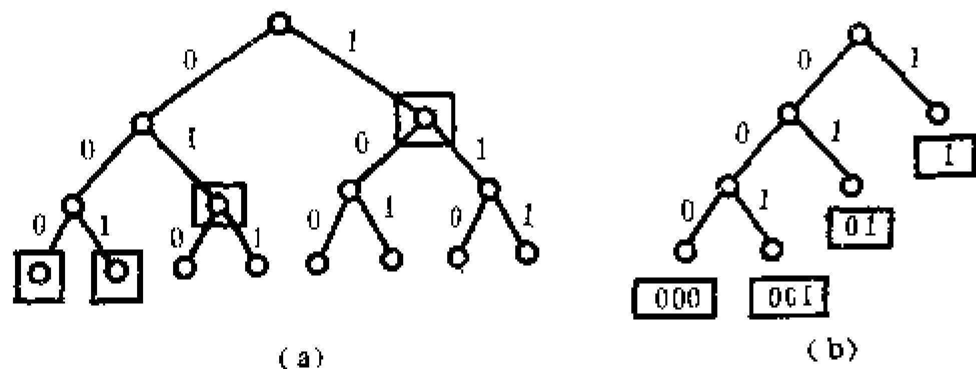


图 3.23

利用完全二元树可以解决字符串长短不同的信息的译码问题。字符串的长短是根据要求予先编好的，即前缀码是予先编好的，对应地确定了一棵完全二元树，当接收到一长串字符时，利用给定的这棵二元树就可进行译码，方法是从树根开始，按着字符的序列前进，如果是 0 就走左边树枝，是 1 就走右边树枝，当走到树叶时，前缀码的第一个序列就被检测到了，然后回到树根重复上述过程寻找下一序列，这样就能准确地将一连串字符分割成前缀码中长短不同的序列。如果字符串的最后部分不能构成前缀码的序列（即从树根往下走还未走到树叶时字符已结束）可约定添加 0 或 1 直到构成序列为止。

例 3.9 设按例 3.8 编制的前缀码及对应的完全二元树，试将接收到的下列信息进行译码：0001100110011101010010。

解：译成：000，1，1，001，1，001，1，1，01，01，001，01，（最后一个字符约定添 1）

至此我们解决了不等长编码的译码问题。然而前缀码是怎样编制的呢？或者说对应的二元树是怎样构造的呢？前面已提到，由于不同字母出现的频率不同，一般希望经常出现的字母的码短些，因此最佳编码的标准就是使得下面码的长度的数学期望

$$L=\sum_{i=1}^{n} w_i\cdot L(v_i) \tag{3.6}$$

取最小值。式中 $L(v_i)$ 表示码 v_i 的长度，w_i 表示它出现的几率，n 是前缀码的基数。

可见设计最佳编码问题也就是寻找最优树的问题，这时树叶表示一个码（字母），它的权表示字母出现的几率，权最小就是出现几率最小的字母，它的通路长度最长，n 就是树叶的数目。因此最优树解决了最佳编码问题。

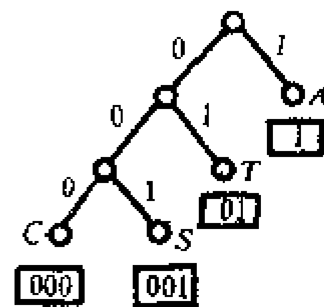


图 3.24

例 3.10 设由 4 个字母A，C，S，T组成的字符，字母出现的几率为A：40%，C：10%，S：20%，T：30%，试编一个相应的二元前缀码并对收到的下列信息进行译码

000100101　000101001　001101　101　0110011

解：构造一棵最优树，它有 4 片树叶A，C，S，T，并分别带权 4，1，2，3，如图 3.24。用最优树对接收的信息泽码，得 CAST CATS SAT AT TASA

二、判定树与搜索树

树的一个重要应用就是用作判定。现以第一章习题 3 为例，将它写出并分析如下：

例 3.11　现有 4 件产品，其中有一件不合格，它的外形并无差异，只是重量不合标准，问如何以最少的次数用天秤找出这件不合格产品，并判别出它的重量比标准产品重还是轻。

解：分别用 a，b，c，d 表示四件产品，构造图 3.25 的一棵树，a：b 表示 a 与 b 在天秤上作比较，"<"表示 a 比 b 轻，树叶即是判定的结论。

上面把作出判定的逻辑关系用树的形式表示出来，使得步骤清晰，一目了然，因而称为判定树。当然构造判定树的方法也不是唯一的，上面的例子也可采取另一种比较方法，因而构造出另一种形式的判定树。

用树进行过程的搜索也是一种常用的方法，这种树称为搜索树，它可以使变化复杂的过程变得条理清晰，从而便于找到有效的方法。下面也举一个例子来说明。

例 3.12　现有 6 根火柴，选手 a、b 轮流从中取走 1 根或 2 根，不能不取，也不许多取，谁取走最后一根他就是胜利者。

解：　以初始状态（6 根火柴）为树根构造一棵二元树，设 a 先取，他可取 1 根或 2 根，因而有二个儿子。树中 a 所面临的分枝点用□表示，b 所面临的分枝点用○表示，[6] 表示轮到 a 取时有 6 根火柴，④表示轮到 b 取时有 4 根火柴，余类推，因此一当出现 [1] 或 [2] 时，a 可一次取走而获胜，同理①或②则是 b 获胜的结局，它们都作为树的树叶，边上的数字表示取走的火柴数。如图3.26。

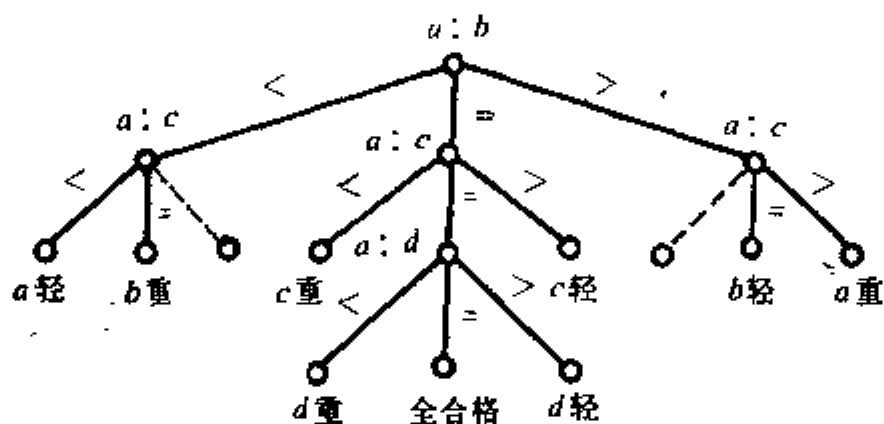


图 3.25

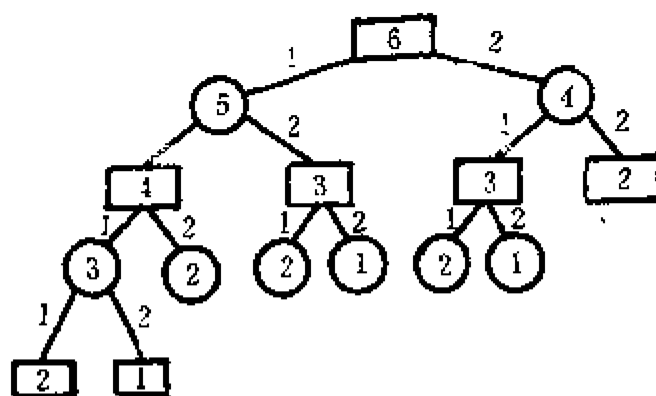


图 3.26

从图可以看出，开始有 6 根火柴，则先取者必败，除非对方失误。

§3.5　树的存贮结构

树在计算机内存中可以通过多重链表来表示，我们把有两个以上指针域的链表称为多重链表。多重链表既可以用来表示线性结构，也可用来表示非线性结构，树就是一种非线

性结构。用多重链表表示一棵树，每个链结点的域的个数依赖于树中该结点的儿子树，结点的一般表示为

DATA	儿子 1	儿子 2	…	儿子 n

由于每个结点的儿子数不等，因而将得到不定长的链结点，给运算增添麻烦，如果取定长的链结点的链表表示树，运算得到简化，但又会浪费较多的存贮空间，前面我们已讲过，任何一棵m元树都可以化成相应的二元树，如果用二元树作为存贮结构，则内存的存贮空间可以节省很多。

对于二元树我们规定：①二元树可以是空的，②二元树的子树有左右之分（左儿子与右儿子）③用有序树表示二元树。于是我们可以得出二元树的一些性质如下：

（1）第 i 层次的结点数最多为 2^{i-1}，$i\geq1$。

（2）高度为 h 的二元树结点总数最多为 $2^{h-1}-1$。

（3）对任一二元树，如果树叶的数目为 n_0 引出次数为 2 的结点数为 n_2，则有 $n_0=n_2+1$。

（4）对于 n 个结点的完全二元树，当按上面规定对结点进行编号后，则对任一结点 i，$1\leqslant i\leqslant n$，我们有

如果 $i\neq1$，则其父亲是 $\left\lfloor\frac{i}{2}\right\rfloor$；如果 $i=1$，则 i 是树根，无父亲。（$\lfloor x\rfloor$ 表示小于等于 x 的最大整数）

如果 $2i\leq n$，则 i 的左儿子是 $2i$，如果 $2i>n$，则 i 无左儿子。

如果 $2i+1\leq n$，则 i 的右儿子是 $(2i+1)$，如果 $2i+1>n$，则 i 无右儿子。

上面的性质用图 3.27 得到说明，证明从略。

一棵二元树可以用两个一维数组 LCHILD 和 RCHILD 来表示，设二元树的结点按 1 至 n 的顺序编号，LCHILD[i]$=j$ 当且仅当 j 是 i 的左儿子，如果 i 没有左儿子，则 LCHILD[i]$=0$，同理可以定义数组元素 RCHILD[i]。

例 3.13 图 3.28（a）（b）给出了一棵二元树和它的数组表示。

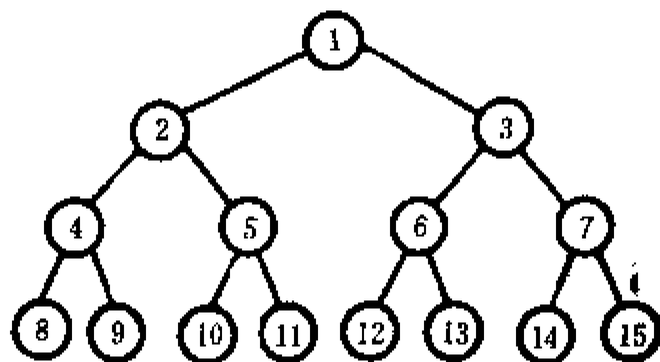
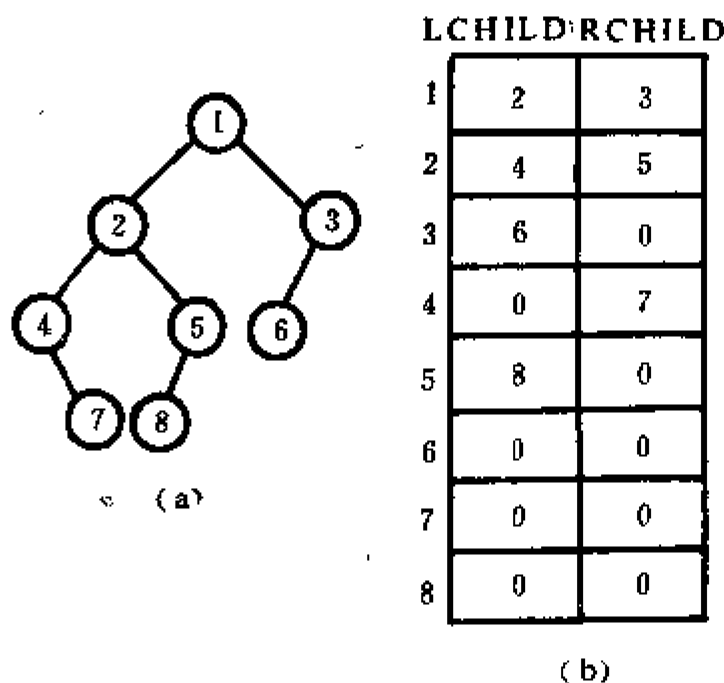


图 3.27



	LCHILD	RCHILD
1	2	3
2	4	5
3	6	0
4	0	7
5	8	0
6	0	0
7	0	0
8	0	0

(b)

图 3.28

对于完全正则二元树可用一个一维数组来表示，既不浪费内存单元，又可以很快地确

定结点的位置，是很方便的。然而数组表示具有一般顺序表示的缺点，即当要插入或删除一个结点时需要移动其他结点，这是很不方便的。为此可以用链表表示二元树，这里每个结点设置三个域：LCHILD, DATA, RCHILD 如图 3.29。

但这种结点结构较难给出其父亲，如要知道任一结点的父亲，还得增加一个父亲域。图 3.30 给出了一般二元树的链表表示法。

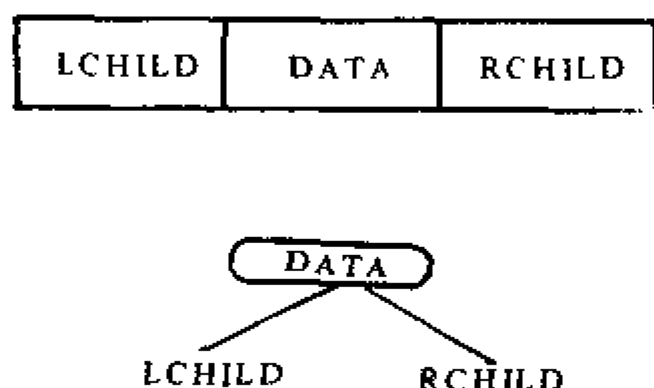


图 3.29

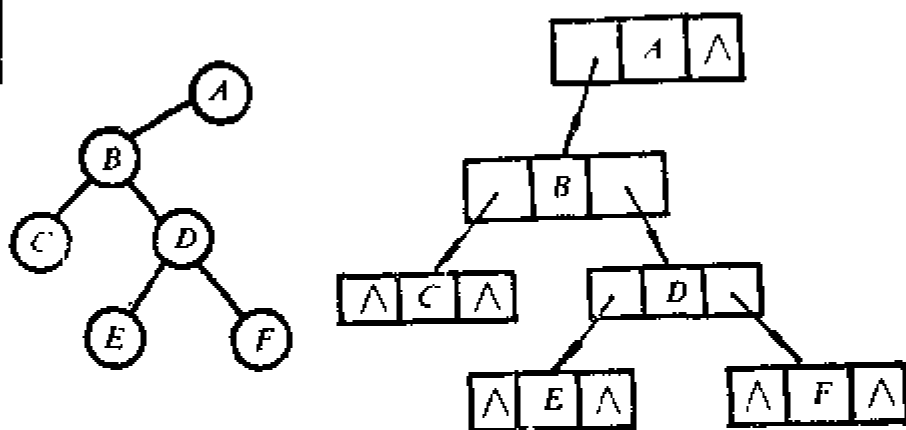


图 3.30

用链表表示二元树也会浪费一些内存单元，当二元树有较多空的左或右子树时更是如此。

结合二元树的存贮，顺便在这里介绍二元排序树的构造及其应用。

二元排序树是树最简单的应用，它的构造步骤如下：设 $R=\{R_1, R_2, \cdots, R_n\}$ 为一数列，按如下原则构造二元排序树

（1）以 R_1 为二元树的根，按数列顺序从 R_2 开始逐个操作。

（2）如果 $R_2<R_1$，则令 R_2 为 R_1 的左子树的根结点，否则令 R_2 为 R_1 的右子树的根结点。

（3）对以后各数递归重复步骤（2），直到 R_n 排完为止。

例 3.14 设 $R=\{20, 5, 3, 4, 15, 2, 30, 25, 18, 40, 22, 10, 35, 45\}$，则按上述步骤构造得到的二元排序树如图 3.31 所示。

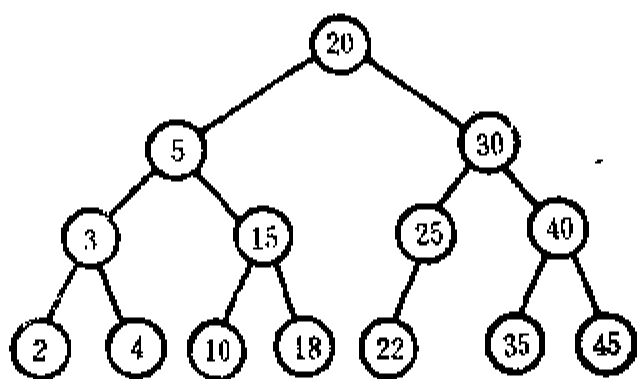


图 3.31

二元排序树的特点是用非线性结构来表示一个线性有序表，下一节我们将看到，对二元排序树遍历访问各结点，就会得到从小到大依次排列的数列。

与一般二元树一样，在内存中二元排序树可用链表来表示，从存贮的角度看，它比用向量表示要占用较多的存贮空间，但它却有一个很大的优点，就是插入一个元素或从原有数列中删除某一元素都很方便，对查找某元素是否在数列中也很方便，下面我们介绍插入，删除与查找这三种操作。

已知一棵二元排序树 T，如果要将某一元素 a 插入树中，按排序树的建立方法，将 a 输入后，a 将排到一片新增加的树叶位置上，将 a 插入时，并不需要移动其他结点，也不需要访问全部结点。

如果要删除二元排序树中的某个结点，也只需移动少量的结点，并且结点删除后仍能保持排序的特性。下面介绍删除结点的算法。

设P是要删除的结点，f是它的父亲，S为删除P后占其位置的结点（即S代替P成为f的儿子），q为结点变量，算法步骤如下。

（1）如果P的左儿子为空，则s←P的右儿子，转（7），否则进行下一步

（2）如果P的右儿子为空，则s←P的左儿子，转（7），否则进行下一步

（3）q←P，s←P的左儿子

（4）当s的右儿子非空时，反复进行

q←s，s←s的右儿子，

直到s的右儿子为空时进行下一步

（5）s的右儿子←P的右儿子

（6）如果q≠P则

q的右儿子←s的左儿子

s的左儿子←P的左儿子

（7）如果P是f的左儿子，则f的左儿子←s，否则f的右儿子←s

（8）结束。

图3.32（a），（b）），（c）分别表示从原有树中删除结点2，6，30的示例。

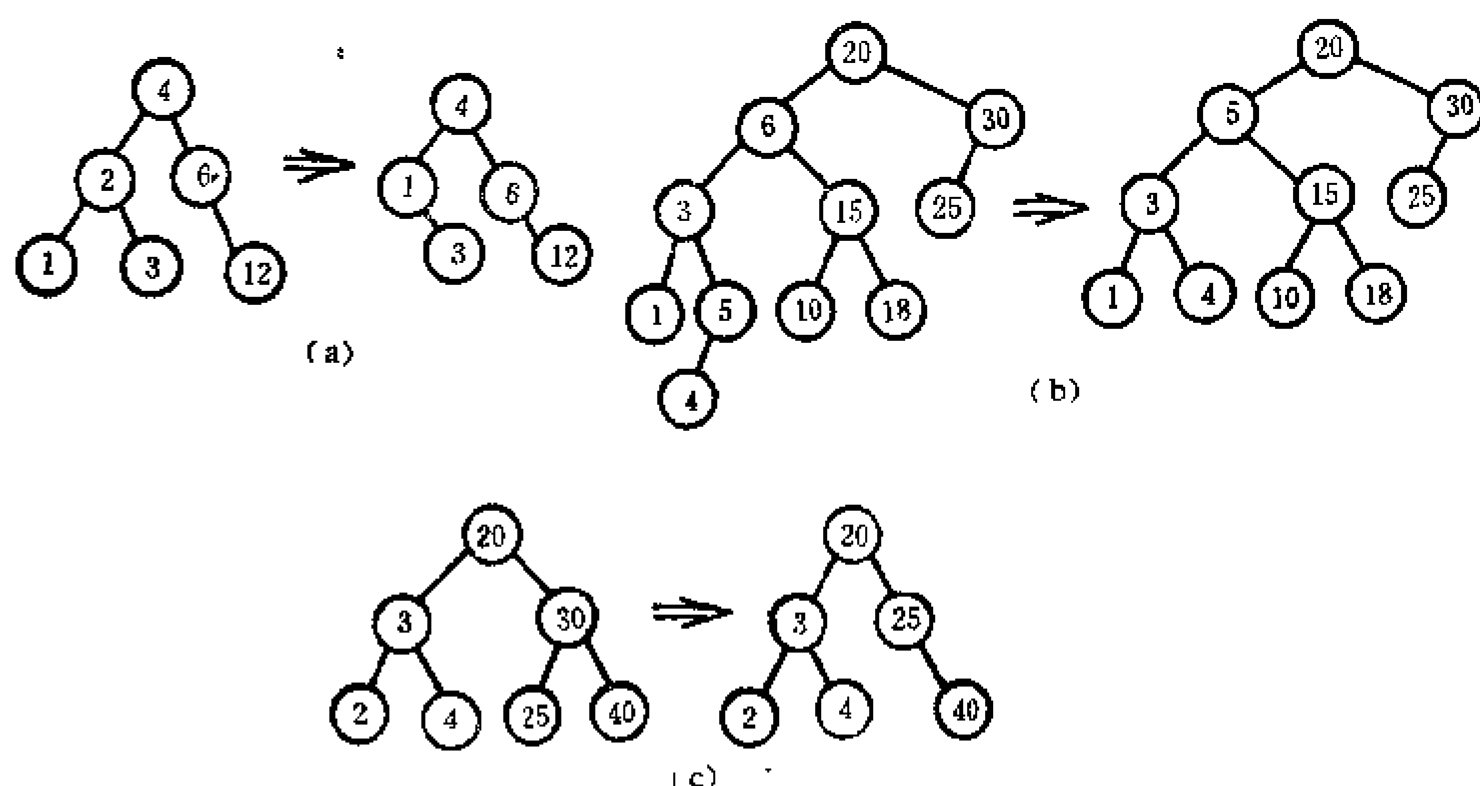


图 3.32

二元排序树的查找过程是一个递归过程，算法步骤如下。

设排序树的根结点为v，要查找的元素为a，查找步骤为

（1）如a=v，输出是“是”，停止，否则进行下一步

（2）如a<v，则

如v有左儿子u，则v←u转（1）否则输出“非”，停止。

否则转下一步（即如a≮v时转下一步）

（3）如v有右儿子w，则v←w，转（1），否则输出“非”，停止。

图 3.33（a），（b）表示查找 55 和71的过程。

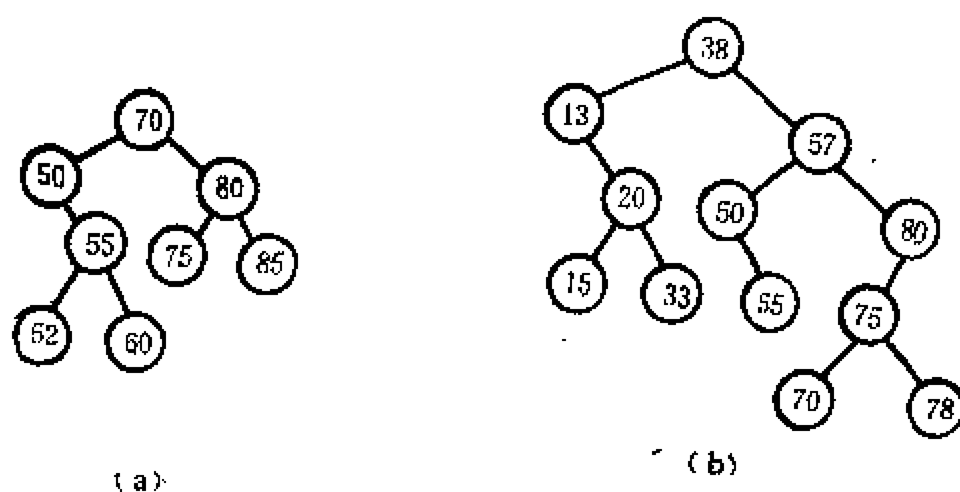


图 3.33

现以图 3.33（a）为例说明查找过程如下：

（1）因a （55）≠v（70），进行下一步

（2）a<v，且v有左儿子u （50），则 v←u（50）转（1）

（1）因a （55）≠v （50），进行下一步

（2）因a （55）≮v （50），转下一步

（3） v （50）有右儿子w （55），则 v←w(55)，转（1）。

（1）因a(55)=v(55)，输出“是”(即树中有这个元素)。

§3.6 树的遍历

很多利用树形结构的算法，常常需要按某种顺序遍历一棵树，即访问每个结点一次仅一次。对于二元树，通常有三种遍历树的方法，分别称为前序遍历，中序遍历与后序遍历。三种遍历都用了递归定义，现分述如下

一、前序遍历（DLR 遍历）

前序遍历的操作为

（1）先访问树根。

（2）在根的左子树上进行前序遍历。

（3）在根的右子树上进行前序遍历。

因此遍历过程是一个递归过程。

例 3.15 对图 3.34作前序遍历。

首先访问树根 1，然后在 1 的左子树上进行前序遍历，即在遍历 1 的左子树时，首先访问树根 2，然后前序遍历 2 的左子树，首先访问树根 3，然后前序遍历 3 的左子树，因 3 的左子树为空，于是前序遍历 3 的右子树，因 3 的右子树也为空，于是前序遍历 2 的左子树结束，开始前序遍历 2 的右子树，先访问右子树的树根 4，然后前序遍历 4 的左子树，先访问树根 5，由于 5 的左右子树均为空，遍历 4 的左子树结束，开始前序遍历 4 的右子树，先访问树根 6，遍历它的左子树 7，遍历它的右子树 8，至此遍历 4 的右子树结束，从而遍历 2 的右子树也结束，于是遍历 1 的左子树亦结束，开始前序遍历 1 的右子

树，先访问树根 9，再遍历 9 的左子树，访问树根10，于是遍历 9 的左子树结束，开始遍历它的右子树，访问树根11，由于11的左右子树均为空，于是遍历 9 的右子树结束，因而遍历 1 的右子树即结束，整个递归过程结束，得到遍历过程结点的访问次序为

1，2，3，4，5，6，7，8，9，10，11

例 3.16 对图 3.35 的二元树进行前序遍历得到序列如下：

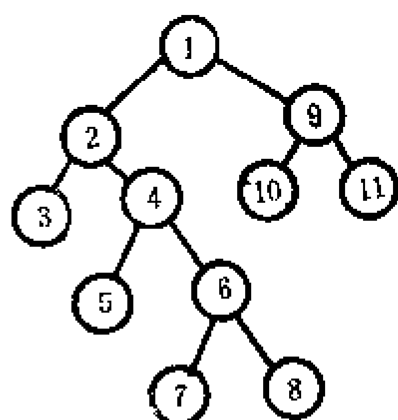


图 3.34

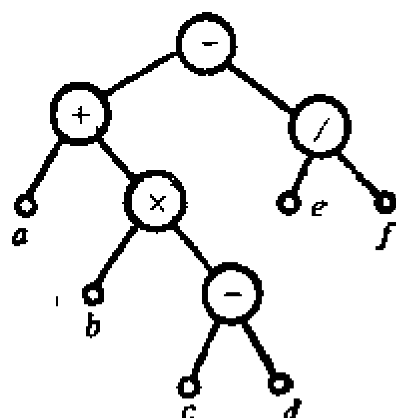


图 3.35

$$-+a\times b-cd/ef$$

这就是算术表达式的前缀表示法，也称波兰表示法，它的运算规则是对算符后面两个紧邻的字符进行运算，写成普通算术表达式即为

$$(a+b\times(c-d))\ -e/f$$

二、中序遍历（LDR）

中序遍历法的操作为

（1）在根的左子树上进行中序遍历。

（2）访问树根。

（3）在根的右子树上进行中序遍历。

例 3.17 对图3.34作中序遍历。

先在根 1 的左子树上作中序遍历，左子树的根结点为 2，于是先在 2 的左子树上作中序遍历，2 的左子树的根结点为 3，于是先在 3 的左子树上作中序遍历，3 的左子树为空，随即访问根结点 3（输出 3），继而在 3 的右子树上作中序遍历，因 3 的右子树为空，于是遍历 2 的左子树结束，随即访问根结点 2（输出 2）继而在 2 的右子树上作中序遍历，因 2 的右子树的根结点为 4，因此先在 4 的左子树上作中序遍历，4 的左子树的根结点为 5，因而又先在 5 的左子树上作中序遍历，5 的左子树为空，随即访问根结点 5（输出 5），继而在 5 的右子树上作中序遍历，5 的右子树为空，于是在 4 的左子树上的中序遍历结束，访问树根 4（输出 4），再在 4 的右子树上作中序遍历。以后的过程都是这样递归地进行，遍历结束，得到的输出为

3，2，5，4，7，6，8，1，10，9，11

例 3.18 对图 3.35 的二元树进行中序遍历得到序列如下：

$$a+b\times c-d-e/f$$

这就是算术表达式的中缀表示，它与普通算术表达式的顺序是一致的，但是括号没有了，因此中缀表示法对表达式的运算来说是不明确的。

不过，如果一个数列采用二元排序树的存贮结构，那么对它进行中序遍历，就会得到一个由小到大顺序排列的数据，故中序遍历仍然具有它的实用价值。例如对图 3.31 的二元排序树进行中序遍历，输出即为

2，3，4，5，10，15，18，20，22，25，30，35，40，45

三、后序遍历（LRD）

后序遍历的操作为

（1）在根的左子树上进行后序遍历。

（2）在根的右子树上进行后序遍历。

（3）访问树根。

例 3.19 对图 3.35 的二元树进行后序遍历得到序列如下：

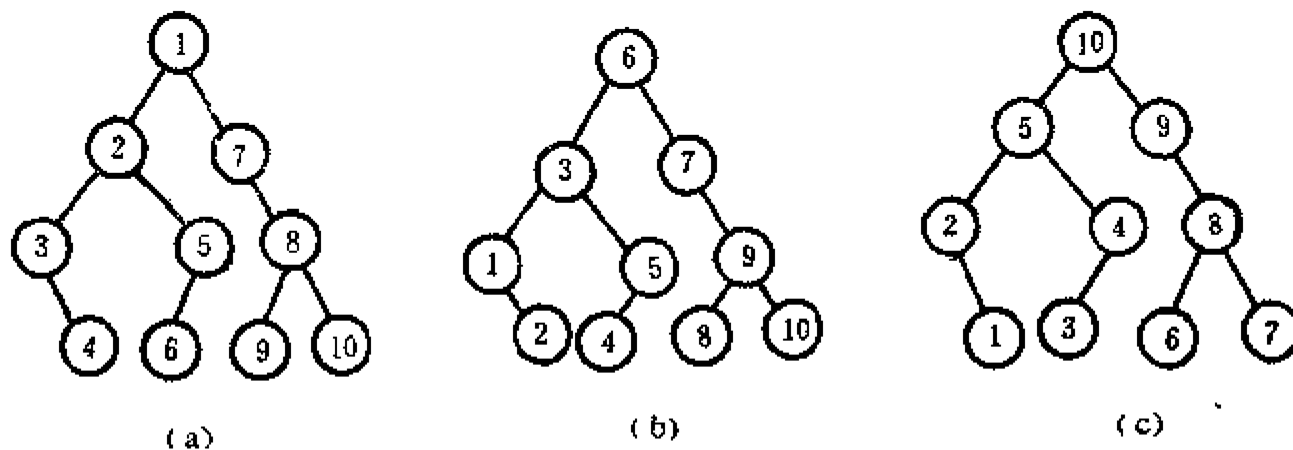
$$abcd-\times+ef/-$$

这就是算术表达式的后缀表示法，也称逆波兰表示法，这种表示法对表达式的编译是很方便的。它的运算规则是对算符左边两个紧邻的字符进行运算，写成普通算术表达式即为

$$(a+b\times(c-d))\ -e/f$$

可见与前序遍历所得到的算术表达式是一致的，只是运算顺序正好相反。

图 3.36 (a)，(b)，(c) 分别给出了按前序、中序及后序三种方式遍历一棵树的结果。其中结点的编号是按访问的先后顺序从小到大给出的。从图中可以看出三种遍历方式的不同。



(a) (b) (c)

图 3.36

一棵树按某种遍历方式给各结点指定了顺序编号，以后用指定的结点编号描述结点是方便的，它能反映出结点之间的某些关系。例如，按前序遍历树的编号中，对任一个以结点 v 为根的子树，根的编号小于它的所有子孙的编号。又如按中序遍历树的编号中，结点 v 的所有左子树上结点的编号都小于 v 的编号，而 v 的所有右子树上结点的编号都大于 v 的编号，这就是对二元排序树进行查找操作时的一个依据。

下面给出一个按中序遍历编号的非递归算法。

这时需要建立一个栈 STACK，并要附设一指针 top，设用向量 S 表示栈，元素尚未进栈时设 top 为零，则 $S[1]$表示第一个进栈的元素，$S[i]$表示第 i 个进栈的元素。算法用两个数组表示二元树，$L[i]$是结点 i 的左儿子的旧编号，$R[i]$是结点 i 的右儿子的旧编号，用向量 count 给各结点编以新号。设二元树的根结点为 v。$n[i]$是旧编号为 i 的结

点的中序遍历编号数。

```
1. procedure  TRAVELTREE
   begin
2. count←1
3. top←1
4. while  L[v]≠0  do
   begin
5. s[top]←v
6. v←L[v]
7. top←top+1
   end
8. n[v]←count
9. count←count+1
10. if  R[v]≠0  do
    begin
11. v←R[v]
12. goto  4
    end
13. if top>0  do
    begin
14. v←s[top]
15. top←top-1
16. goto  8
    end
    end
```

§3.7 图的中心和中位点

图的中心和中位点，与研究选址问题有密切关系。我们知道，图是运输网、通讯网、公用事业服务网等等的数学模型，在研究各种网络时，都存在怎样给某些设施选择最优地址的问题。例如一个县的各个乡镇之间有公路连接，形成了一个交通网，现在拟在某个镇上设立一个急救站，要求急救站到最远的乡镇距离达到最小，急救站应设在哪个镇上才能满足这个要求，这就是图的中心问题。又如这个县要设立一个邮局，每天把邮件分送到县属各个乡镇，邮局设立在哪个乡镇上，才使每天送邮件的总行程最小，这又是图的中位点问题。

评价一个系统的质量，由于系统的功能不同，也有不同的标准，但中心和中位点问题却是普遍关心的。下面分别就这两个问题进行讨论。

一、图的中心

定义 3.18 设G是有n个结点的连通图。

（1）结点 v_i 与其最远点的距离称为 v_i 的半径，记作 ρ_i，即

$$\rho_i=\max_{1\leq j\leq n}\{d(v_i,v_j)\} \tag{3.7}$$

（2）半径最小的结点称为图的中心，记作 v_c，其半径称为图的半径，记作 ρ_G，即

$$\rho_G=\max_{1\leq j\leq n}\{d(v_c,v_j)\}$$

$$=\min_{1\leq i\leq n}\left\{\max_{1\leq j\leq n}[d(v_i,v_j)]\right\} \tag{3.8}$$

式（3.8）的涵义为与最远点的距离最小的点就是图的中心。

图3.37表示的无向图，从直观可见 v_5 是图的中心。一个图可以不止一个中心，如图3.38就有三个中心，即 v_1，v_3和 v_5。

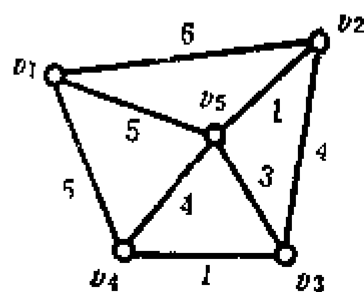


图 3.37

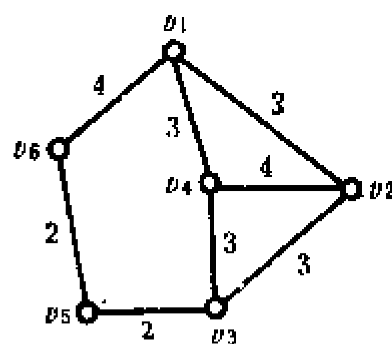


图 3.38

常把图的任意两点之间距离的最大值，称为图的直径。一般情况下图的直径并不是它的半径的二倍。

由图的距离矩阵很容易求出图的中心及其半径，步骤如下

（1）求出图 G 的距离矩阵 $D=(d_{ij})_{n\times n}$

（2）对每一 i，$1\leq i\leq n$，

$$\rho_i=\max_{1\leq j\leq n}\{d(i,j)\}(\rho_i\text{为结点 } v_i \text{ 的半径})$$

（3）

$$\rho_G=\min_{1\leq i\leq n}\left\{\rho_i\right\}=\max_{1\leq j\leq n}\left\{d(k,j)\right\}$$

$$v_c=v_K$$

例 3.20 求图3.38的中心和半径

首先求出它的距离矩阵

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	3	6	3	6	4
v_2	3	0	3	4	5	7
v_3	6	3	0	3	2	4
v_4	3	4	3	0	5	7
v_5	6	5	2	5	0	2
v_6	4	7	4	7	2	0

$D=$ 上述矩阵

易见 $\rho_1=\max\{0,3,6,3,6,4\}=6$

$\rho_2=7$

$\rho_3=6$

$\rho_4=7$

$\rho_5=6$

$\rho_6=7$

所以图的半径 $\rho_G=6$，图的中心为：v_1，v_3，v_5。

式（3.7）和（3.8）不仅适用于无向图，亦适用于有向图或混合图，请看下例。

例 3.21 求图 3.39 的中心和半径。

先求出图的距离矩阵

$$D=\begin{matrix} & v_1 & v_2 & v_3 & v_4 \\ v_1 & 0 & 2 & 3 & 3 \\ v_2 & 4 & 0 & 2 & 1 \\ v_3 & 6 & 2 & 0 & 3 \\ v_4 & 3 & 5 & 4 & 0 \end{matrix} \qquad \begin{matrix} \rho_1=3 \\ \rho_2=4 \\ \rho_3=6 \\ \rho_4=5 \end{matrix}$$

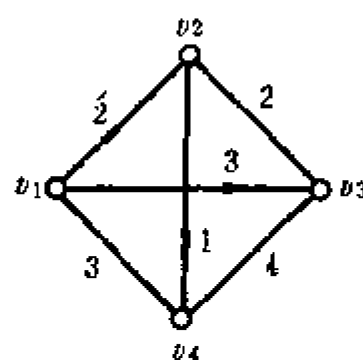


图 3.39

所以图的半径 $\rho_G=3$，图的中心是点 v_1。

图的中心和半径的定义也适用于树，当系统的结构是树时，也存在寻求树的中心的问题。一般而言，一个图可以有很多中心，例如成为一个回路的多边形，每个点都是中心，可是树却不同，有如下定理。

定理 3.13 每棵树只有一个或两个中心。

证：树的结点分两类，一类是树叶，另一类是分枝点，树叶的半径大于分枝点的半径，因此树的中心一定是分枝点而不会是树叶（$n>2$时）。现在我们来分析分枝点的半径，用图 3.40（a）表示一棵树T，将T的所有树叶及其关联边都去掉，树T变成树T'，如图（b），可以看出，T'中各分枝点的半径的值与T中对应分枝点半径的值相比较，都普遍地减少 1，因此T的中心点仍然是T'的中心点。再将T'的所有树叶及其关联边都去掉得T''，如图（c），则T'的中心点也是T''的中心点，如此继续下去，最后只剩下一个结点，如图（d），它就是树的中心。也可能最后剩下一条边，如图 3.41的树，它有两个中心。 ▮

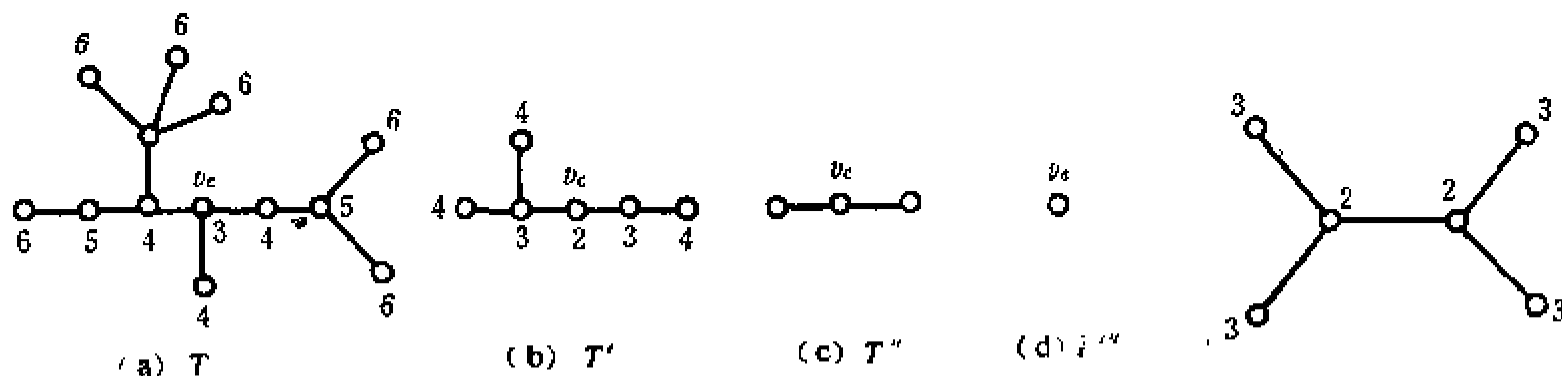


图 3.40　　　　图 3.41

图 3.40 和图 3.41 中结点上标的数字为结点的半径。

推论：如果一棵树有两个中心，这两个中心必是相邻的。

二、图的中位点

定义 3.19 设G是有n个结点的连通图

（1）结点v_i与其余结点距离之和称为v_i的总线长，记作S_i，即

$$S_i=\sum_{j=1}^{n} d(v_i, v_j) \qquad (3.9)$$

（2）总线长最短的结点称为图的中位点，记作v_m，其总线长称为图的中线长，记作S_G，即

$$S_G=\sum_{j=1}^{n} d(v_m, v_j)=\min_{1\leqslant K\leqslant n}\sum_{j=1}^{n} d(v_K, v_j) \qquad (3.10)$$

式（3.10）的涵意为与所有各点距离之和为最小的结点就是图的中位点。

与求图的中心方法类似，由图的距离矩阵很容易求出它的中位点，举例说明如下

例 3.22 如例 3.21，求图的中位点。

由图的距离矩阵可得

$$S_1 = 0 + 2 + 3 + 3 = 8$$

$$S_2 = 4 + 0 + 2 + 1 = 7$$

$$S_3 = 6 + 2 + 0 + 3 = 11$$

$$S_4 = 3 + 5 + 4 + 0 = 12$$

所以 $S_G = \min\{8, 7, 11, 12\} = 7 = S_2$，$v_2$ 是图的中位点。

在一些实际系统中，还存在加权选址问题，举例如下。

例 3.23 某企业有 5 个分点，它们之间的交通路线如图3.42，必须把产品的总和$X=(50, 75, 30, 100, 25)$分送到 5 个点，设运费$C_t=2$（单位距离、重量运费），问企业应设在哪个分点可使运费最低。

解：先求出图的距离矩阵如下：

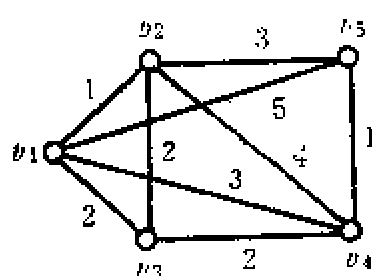


图 3.42

$$D = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 2 & 4 & 3 \\ 2 & 2 & 0 & 2 & 3 \\ 3 & 4 & 2 & 0 & 1 \\ 4 & 3 & 3 & 1 & 0 \end{bmatrix} \end{array}$$

设 Q_i 为企业设在点 v_i 时运输费总和，则

$$Q_1 = (1\times75+2\times30+3\times100+4\times25)\times2=1070$$

$$Q_2 = (1\times50+2\times30+4\times100+3\times25)\times2=1170$$

$$Q_3 = (2\times50+2\times75+2\times100+3\times25)\times2=1050$$

$$Q_4 = (3\times50+4\times75+2\times30+1\times25)\times2=1070$$

$$Q_5 = (4\times50+3\times75+3\times30+1\times100)\times2=1230$$

最小值为 Q_3，故企业应设在 v_3 上。

无论寻找图的中心或中位点，算法中都需要先求出图的距离矩阵D，D的涵意是不难理解的，至于如何求一个图的距离矩阵，我们将在第七章§7.2中讨论。

§3.8 图的块划分

图的连通性是图的重要特征之一，上一章§2.2中已给出图的连通性定义，并在§2.3中讲了用图的邻接矩阵和可达性矩阵判别有向图连通类的方法。这一节我们主要讨论无向图的连通性及其算法，并给出一个求有向图强连通块的较好算法。

一、无向图的连通性

分析一个无向图，往往首先就要判定图是否是连通的，如果不是连通的，有多少个连通分支。

判定无向图的连通与否，方法是比较多的，上一章讲图的遍历，实质上也是对图的连通性判断，无论是深度优先或是广度优先搜索法，当边数 $i=n-1$ 时，说明遍历的过程形成了一棵生成树，因而可知图是连通的，而且我们还给出了一个可以判定不连通无向图的遍历算法，这里不再重复。

给出一个图的邻接矩阵 A，如果对 A 施行行列置换之后，能得到下面的形式，就可以确定图的连通分支，但是这是一个效率差的方法，因为可能要进行 $n!$ 次置换。

$$A=\begin{bmatrix} A_{11} & \bigcirc \\ \bigcirc & A_{22} \end{bmatrix}$$

另一个方法是仿照求有向图的可达性矩阵，求出无向图邻接矩阵的各次幂（用布尔算法），再求出它们的布尔和，即

$$P=A\vee A^{(2)}\vee\cdots\vee A^{(n)}$$

从 P 即可判断图的连通性，这种算法比较麻烦，但是可达性矩阵可以采用 Warshall 算法，在这一算法的启发下，对无向图的连通性判定，可以采用下一算法。

算法的基本思路是将邻接结点合并。从图的某一结点 v_0 出发，所有与 v_0 邻接的结点都是彼此连通的，将这些结点与 v_0 合并成一个新的结点 u_0，于是凡是与合并在 u_0 中的结点邻接的结点，都看成与 u_0 邻接，于是又可把它们合并成一个新的结点。如此继续下去，最后合并得的新结点 V_1 如果不再与外面的任一结点邻接，则 V_1 内的那些结点就形成一个连通分支。如果 $V_1=V$，则图是一个连通图，否则从图中删去 V_1 中的结点及其关联边，得到图 G 的一个子图 G'，再对 G' 重复上述步骤，又可得到 G 的另一连通分支，由于图的有限性，最后即可求出图的分支数及各分支子图。

上述方法可在图的邻接矩阵上进行。把结点 j 与结点 i 合并采用逻辑“或”的操作，即用逻辑加法将行 j 加到行 i，列 j 加到列 i，并删除行 j 和列 j，于是得到一个 $(n-1)$ 阶新矩阵。又重复上述操作，如果新矩阵中所在的行不再有邻接点，表明新结点内的结点已形成一个连通子图，在新矩阵中可将新结点所对应的行和列删除，继续对余下的矩阵运算，如果通过行、列运算最后只剩下一行一列，即成为一个点，表明图是连通的。

通过合并，一个自环在主对角线上出现为 1，而平行边由于逻辑加法自动成为一条边，这些对于图的连通性是没有影响的。

在这个算法中，可能进行的最大合并数为 $n-1$，在每次合并中，最多进行 n 次逻辑加法，因而时间的计算量是 $O(n^2)$

例 3.24 求图3.43的连通子图，已知

$$A=\begin{array}{c|ccccccc} & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \\ \hline v_1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ v_2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ v_3 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ v_4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ v_6 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ v_7 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}$$

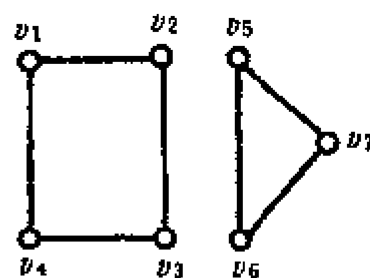


图 3.43

结点的合并过程如下：

$$
A \xrightarrow[\text{合并}]{\text{点}1,2,4}
\begin{array}{c}
\\ (1,2,4) \\ 3 \\ 5 \\ 6 \\ 7
\end{array}
\begin{array}{ccccc}
(1,2,4) & 3 & 5 & 6 & 7 \\
1 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1
\end{array}
\xrightarrow[\text{合并}]{(1,2,4),3}
$$

$$
\begin{array}{c}
\\ (1,2,3,4) \\ 5 \\ 6 \\ 7
\end{array}
\begin{array}{cccc}
(1,2,3,4) & 5 & 6 & 7 \\
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1
\end{array}
\xrightarrow[\text{行,列}]{\text{删去}(1,2,3,4)}
\begin{array}{c} 5 \\ 6 \\ 7 \end{array}
\begin{pmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{pmatrix}
\xrightarrow[\text{合并}]{\text{点}5,6,7}
\begin{pmatrix} 1 \end{pmatrix}
$$

于是得到两个连通子图，它们的结点集合为：$\{v_1,v_2,v_3,v_4\}$，$\{v_5,v_6,v_7\}$。

此外，求带权无向图最小生成树的两个算法，只要稍加改动，也可用来判定无向图的连通性及连通分支，现以 *Kruskal* 算法为例对它稍作改动得到判定无向图连通性的算法如下

1. $VS \leftarrow \varnothing$
2. 将E中的边排成序列Q
3. for 每个结点 $v \in V$ do $VS \leftarrow VS \cup \{\{v\}\}$
4. for i = 1 to $|E|$ do

   begin

5. 从Q中取出边 (v,w)
6. 从Q中删去边 (v,w)
7. if v,w 在 VS 的不同元素集 V_1，V_2 中 then $V \leftarrow V_1 \cup V_2$

   end

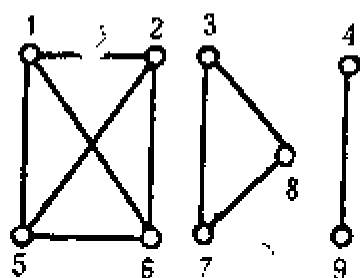


图 3.44

算法中 VS 是一个不相交的结点集合，初始状态时 $VS = \{\{v_1\},\ \{v_2\},\ \cdots,\ \{v_n\}\}$，运算结束如 $|VS| = 1$ 则图是连通的，否则它的值即为图的连通分支数。

例 3.25 用上述算法判定图 3.44 的连通性。

解： 边的序列 Q 为

(1,2)，(3,8)(4,9)，(1,6)，(2,5)，(3,7)，(1,5)，(2,6)，(3,8)，(7.8)，计算步骤列表如下。

步骤	取出边	操作	VS
1	(1,2)	合并点	{{1,2},{3},{4},{5},{6},{7},{8},{9}}
2	(3,8)	同上	{{1,2},{3,8},{4},{5},{6},{7},{9}}
3	(4,9)	同上	{{1,2},{3,8},{4,9},{5},{6},{7}}
4	(1,6)	同上	{{1,2,6},{3,8},{4,9},{5},{7}}
5	(2,5)	同上	{{1,2,5,6},{3,8},{4,9}{7}}
6	(3,7)	同上	{{1,2,5,6},{3,7,8},{4,9}}
7	(1,5)	不作什么	同上
8	(2,6)	同上	同上
9	(3,8)	同上	同上
10	(7,8)	同上	同上

本割集，它们构成图G关于生成树T的基本割集组，记作K_T，即$K_T=\{K_1,K_2,\cdots,K_{n-1}\}$。

这里我们把基本割集说成是图G关于生成树T的基本割集，就是因为基本割集是对应给定的生成树而言的，显然不同的生成树具有不同的树枝，因而对应的基本割集也不同，将得到不同的基本割集组，但是基本割集的数目是相同的，而且基本割集的性质也不因生成树选择的不同而有所差异，所以我们可以选择任意一棵生成树对应的基本割集进行分析，并且把它简称为图G的基本割集。

定理 3.27 图G的$n-1$个基本割集$K_1,K_2,\cdots,K_{n-1}$是线性无关的。

证：因为每一个基本割集只含一条树枝，不同的基本割集含不同的树枝，所以基本割集的环和不可能是空集。 ▌

定理 3.28 图G的任一断集均可表示为若干个基本割集的环和。

证：假设选定了某一棵生成树，得到一组基本割集。并设S_t是图G的任意一个断集，则S_t至少包含生成树的一条树枝，不妨设

$$S_t=\{e_1,e_2,\cdots,e_i,e_{i+1},\cdots,e_j\}$$

其中$e_1,e_2\cdots,e_i$是树枝，$e_{i+1},\cdots,e_j$是弦。设$K_1,K_2\cdots,K_i$是分别含树枝$e_1,e_2,\cdots,e_i$的基本割集，则

$$S=K_1\oplus K_2\oplus\cdots\oplus K_i$$

是G的一个断集，由环和运算及基本割集的性质可知断集S仅含树枝$e_1,e_2,\cdots,e_i$而不再含其他树枝，但S_t也仅含这些树枝，故$S\oplus S_t$将不含树枝，根据定理3.21，$S\oplus S_t$如仍是断集，不可能不含树枝，因此只有S与S_t是两个相同的断集，才不致产生矛盾，即

$$S_t=S=K_1\oplus K_2\oplus\cdots\oplus K_i$$ ▌

定理3.27和定理3.28表明，基本割集组构成了断集空间的基底，通过基本割集各种可能的环和就可能获得断集空间的全部断集，于是我们又找到了寻求图的全部割集的另一种方法，就是通过基本割集求出所有割集，因此这一方法的关键在于求出基本割集，基本割集只含一条树枝，其余都是弦，怎样找出这些弦，又成为求基本割集的关键。为此我们引入如下定理。

定理 3.29 设T是连通图G的一棵生成树，则含树枝e的基本割集$K(e)$如还含有其他一些边，则这些边是由所有含树枝e的基本回路中的弦所组成。

证：首先证明含树枝e的基本回路中的弦一定是割集$K(e)$中的一条边。用反证法，设含树枝e的某一基本回路的弦a不在割集$K(e)$中，则a的两个端点必都在结点集合V_1中或者都在结点集合V_2中（参看图3.58），设a的两个端点在V_1中，则从V_1经树枝e到V_2之后，必须再经过V_2与V_1之间的一条弦回到V_1才可能与a形成回路，显然这不是基本回路，同理如果弦a的两个端点都在V_2中也不可能构成含树枝e的基本回路，因此含e的基本回路的弦a，它的两个端点只能一个在V_1中，另一个在V_2中，所以a一定是割集$K(e)$的一条边。

其次证明割集$K(e)$中的边除树枝e外，其余都是含e的基本回路的弦。现考察割集$K(e)$中除树枝e外的任一条边e'，因e'不是树枝而是弦，故将e'加到树T上必然产生一个基本回路，因e'是跨接在V_1与V_2之间的弦，必须再经过V_1与V_2之间的一条边才可能形成回路，因V_1与V_2之间只有一条边e是树枝，e'只能经过e形成的回路才会是基本回路，所以$K(e)$中除e之外的边，不仅都是弦，而且都是含树枝e的基本回路

所对应的弦。 ▌

下面给出求图的所有基本回路的一个算法，算法中T表示图G的一棵生成树，CT是T的补，FCS为基本回路的集合。算法的时间复杂性为$O(n^3)$。算法步骤如下。

1）求出G的生成树T及其补CT

2）$FCS \leftarrow \phi$

3）for 所有的边 $e_i=(v_i, v_i') \in CT$ do

begin

4）找出在T中从v_i到v_i'的路经并记作P_i

5）$C_i \leftarrow P_i \cup \{e_i\}$

6）$FCS \leftarrow FCS \cup C_i$

end

例 3.33 求出图3.57所示的无向图的全部割集。

选定图的一棵生成树T如图3.59（a）所示，

$$T=\{e_1, e_5, e_6, e_7\}, \quad CT=\{e_2, e_3, c_4\}$$

用上述算法求出图的全部基本回路如下：

对应弦e_2的基本回路：$C(e_2)=\{e_2, e_5, e_6\}$

对应弦e_3的基本回路：$C(e_3)=\{e_3, e_1, e_5, e_7\}$

对应弦e_4的基本回路：$C(e_4)=\{e_4, e_1, e_5\}$

含树枝e_1的基本回路有$C(e_3), C(e_4)$，则

$$K(e_1)=\{e_1, e_3, e_4\}$$

含树枝e_5的基本回路有$C(e_2), C(e_3), C(e_4)$则

$$K(e_5)=\{e_5, e_2, e_3, e_4\}$$

同理可求 $K(e_6)=\{e_6, e_2\}, K(e_7)=\{e_7, e_3\}$

于是得到基本割集组如下，它们构成断集空间的基底

$$K_T=\{K(e_1), K(e_5), K(e_6), K(e_7)\}$$

求出基本割集的各种环和

$$K(e_1) \oplus K(e_5)=\{e_1, e_2, e_5\}=K_1$$

$$K(e_1) \oplus K(e_6)=\{e_1, e_2, e_3, e_4, e_6\}$$

$$K(e_1) \oplus K(e_7)=\{e_1, e_4, e_7\}=K_2$$

$$K(e_5) \oplus K(e_6)=\{e_3, e_4, e_5, e_6\}=K_3$$

$$K(e_5) \oplus K(e_7)=\{e_2, e_4, e_5, e_7\}=K_4$$

$$K(e_6) \oplus K(e_7)=\{e_2, e_3, e_6, e_7\}$$

$$K(e_1) \oplus K(e_5) \oplus K(e_6)=\{e_1, e_5, e_6\}=K_5$$

$$K(e_1) \oplus K(e_5) \oplus K(e_7)=\{e_1, e_2, e_3, e_5, e_7\}$$

$$K(e_1) \oplus K(e_6) \oplus K(e_7)=\{e_1, e_2, e_4, e_6, e_7\}=K_6$$

$$K(e_5) \oplus K(e_6) \oplus K(e_7)=\{e_4, e_5, e_6, e_7\}=K_7$$

$$K(e_1) \oplus K(e_5) \oplus K(e_6) \oplus K(e_7)=\{e_1, e_3, e_5, e_6, e_7\}$$

通过辨识，$K_1 \sim K_7$都是割集，加上4个基本割集，全部割集共有11个，余下的4个则是断集。与例3.32比较，两种算法的结果完全一致。

通过以上分析及例题，我们可以得出如下几点结论。

（1）n阶连通图G的断集空间的元素共有2^{n-1}个（包含空集）

（2）n阶完全图G共有（2^{n-1}）个割集，其中含（$n-1$）个基本割集。

（3）n个结点的树有（$n-1$）个割集，它们都是基本割集，也是割边。

因此一个n阶连通图最少有（$n-1$）个割集，最多有（$2^{n-1}-1$）个割集。下面简单介绍割集矩阵。

定义 3.28 一个n阶连通图G的割集矩阵记作$Q=(q_{ij})$，其中

$$q_{ij}=\begin{cases}1,\text{若边 } e_j \text{ 在割集 } K_i \text{ 中}\\0,\text{否则}\end{cases}$$

例如图3.60表示的连通图，可以求出它们全部割集如下：

$K_1=\{e_1,e_2,e_4\}$

$K_2=\{e_2,e_3,e_4,e_6\}$

$K_3=\{e_4,e_5,e_6\}$

$K_4=\{e_1,e_3,e_6\}$

$K_5=\{e_2,e_3,e_5\}$

$K_6=\{e_1,e_2,e_5,e_6\}$

$K_7=\{e_1,e_3,e_4,e_5\}$

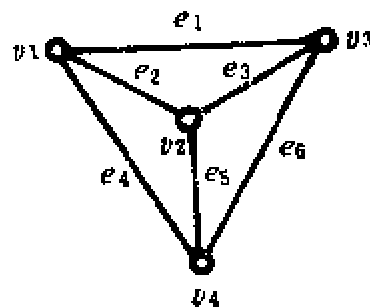


图 3.60

图的割集矩阵为

$$\begin{array}{c} \\ K_1\\K_2\\K_3\\K_4\\K_5\\K_6\\K_7\end{array}\begin{array}{c}\begin{array}{cccccc}e_1&e_2&e_3&e_4&e_5&e_6\end{array}\\ \left(\begin{array}{cccccc}1&1&0&1&0&0\\0&1&1&1&0&1\\0&0&0&1&1&1\\1&0&1&0&0&1\\0&1&1&0&1&0\\1&1&0&0&1&1\\1&0&1&1&1&0\end{array}\right)\end{array}$$

定义 3.29 一个n阶连通图的基本割集矩阵$Q_f=(q_{ij})_{(n-1)\times e}$，其中

$$q_{ij}=\begin{cases}1,\text{若边 } e_j \text{ 在基本割集 } K_i \text{ 中}\\0,\text{否则}\end{cases}$$

例如图3.60所示的无向图，给定生成树$T=\{e_1,e_3,e_5\}$，则基本割集为

$K_1=\{e_1,e_2,e_4\}$

$K_2=\{e_2,e_3,e_4,e_6\}$

$K_3=\{e_4,e_5,e_6\}$

图的基本割集矩阵为

$$Q_f=\begin{array}{c} \\ K_1\\K_2\\K_3\end{array}\begin{array}{c}\begin{array}{cccccc}e_1&e_2&e_3&e_4&e_5&e_6\end{array}\\ \left(\begin{array}{cccccc}1&1&0&1&0&0\\0&1&1&1&0&1\\0&0&0&1&1&1\end{array}\right)\end{array}$$

基本割集矩阵的秩为$n-1$，割集矩阵的行向量不是线性独立的，它的秩也是$n-1$，一般常用图的基本割集矩阵。

习题与思考题

1．一棵树有两个结点的次数为2，一个结点的次数为3，3个结点的次数为4，问它有几个次数为1的结点。

2．一棵树有n_2个结点的次数为2，n_3个结点的次数为3，n_K个结点的次数为K，其余结点的次数均为1，问次数为1的结点有多少。

3．画出有7个结点非同构的树。

4．证明有n个结点的树，结点的次数和为$2n-2$。

5．证明：如T是一棵树，则T中最长通路的起点和终点的次数均为1．

6．证明：具有n个结点及至少n条边的图含有回路。

7．证明：在具有3个或更多个结点的连通图中，至少有2个结点不是割点。

8．试证明：连通无向图的任何一条边，都可以是生成树的树枝。

9．证明或反证下列命题：连通无向图G的任何一条边都是G的某一生成树的弦。

10．设T_1和T_2是连通图G的两棵生成树，a是在T_1中但不在T_2中的一条边。证明：存在一条边b，它在T_2中但不在T_1中，使得$(T_1-\{a\})\cup\{b\}$和$(T_2-\{b\})\cup\{a\}$都是G的生成树。

11．设$G=(V,E)$为连通图，$e\in E$，证明：当且仅当e是G的割边时，e才能在G的任何一棵生成树中。

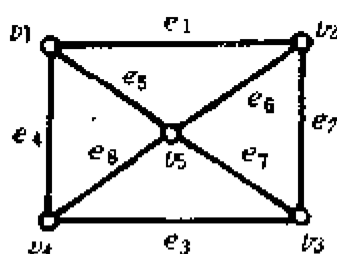


图 3.61

12．如图3.61所示的连通图，求出它的所有回路。

13．如图3.61所示的图G，选取G的一棵生成树，证明G的任一回路至少有一条边是弦。

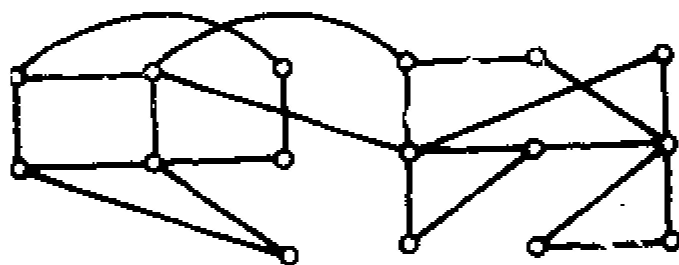
14．证明：完全图任一生成树的弦的集合是一连通子图。

15．设树T的某一结点v的次数$\deg(v)=K$，证明T中至少有K个次数为1的结点。

16．证明：在完全图G中，如G至少有5个结点，则对应于任何生成树的弦的集合至少含有一个回路。

17．图3.62给出了一个无向图，任意求出它的三棵生成树。

18．在图3.63中，取三棵生成树T_1，T_2，T_3，分别计算它们的权并比较其大小。


图 3.62

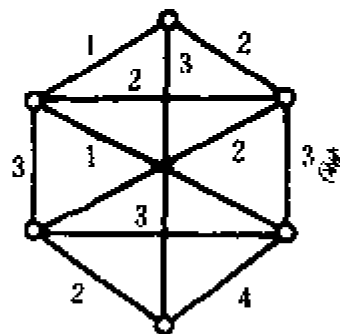


图 3.63

19. 试用 *Kruskal* 算法求图 3.64 的最小生成树。

20. 试用 *Prim* 算法求图 3.65 的最小生成树。

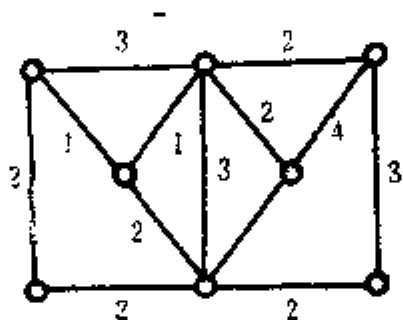


图 3.64

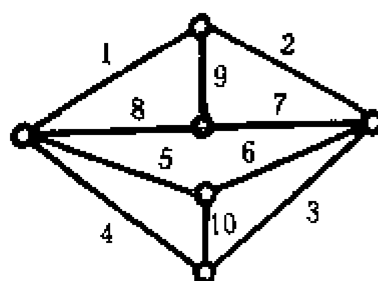


图 3.65

21. 已知世界六大城市：北京（B），纽约（N），巴黎（P），伦敦（L），东京（T），墨西哥（M），试从下表给出的交通网络中求最小生成树。

	B	T	P	M	N	L
B	＼	13	51	77	68	50
T	13	＼	60	70	67	59
P	51	60	＼	57	36	2
M	77	70	57	＼	20	55
N	68	67	36	20	＼	34
L	50	59	2	55	34	＼

单位：百英里

22. 根据简单有向图的邻接矩阵，如何才能确定它是否是棵有向树？如果给定的简单有向图是棵有向树，那么应如何确定它的根和叶。

23. 试举例说明，一个简单有向图，仅有一个结点的引入次数为 O，其余结点的引入次数均为 1 不一定是有向树。

24. 由三个结点能够构成多少种不同的树？有向树和有序树？

25. 证明在完全二元树中，边的总数等于 $2(K-1)$，其中 K 是树叶的数目。

26. 证明在完全二元树中有奇数个结点。

27. 在一棵完全 t 元树中其外部通路长度与内部通路长度之间有什么关系。

28. 求出图3.66对应的二元树。

29. 求出图3.67所表示的森林所对应的二元树。

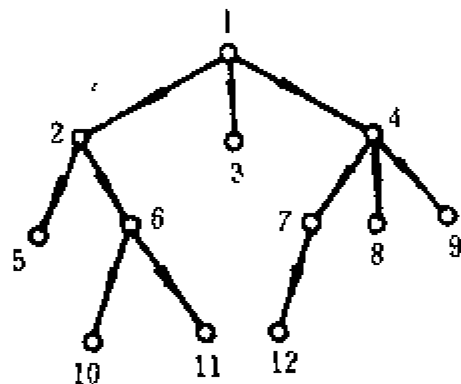


图 3.66

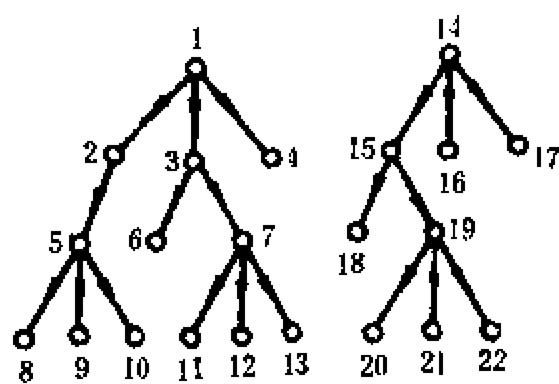


图 3.67

30．将图3.68（a），（b）的二元树转化成对应的m元树。

31．设v是树T的一个结点，给T的树枝加上一个方向，可以得到一棵以v为树根的有向树。设$v_1,v_2,\cdots,v_K$表示v的儿子，$S_1,S_2,\cdots,S_K$表示以$v_1,v_2,\cdots,v_K$为树根的子树中结点的个数，点v的权定义为$S_1,S_2,\cdots,S_K$中的最大值。

（1）对于图3.69所示的树，计算各结点的权。

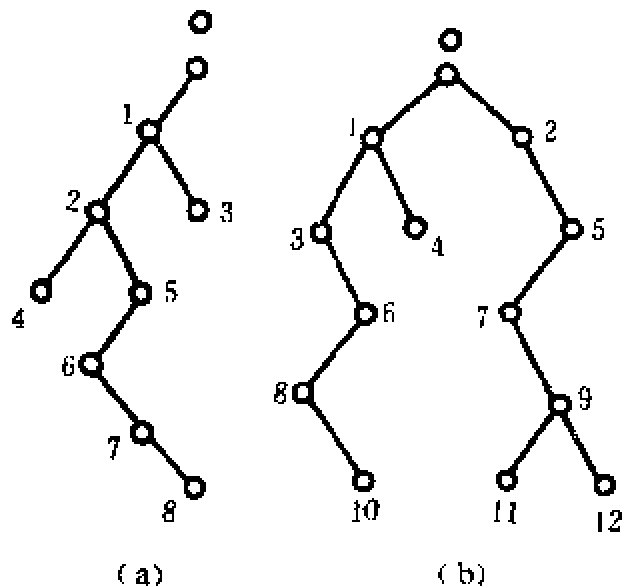


图 3.68

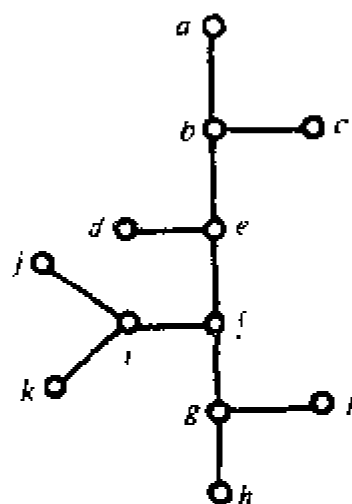


图 3.69

（2）具有最小权的结点称为树的质心，试确定这棵树的质心。

（3）证明一棵树有一个或两个质心，而且如果有两个质心，它们一定是相邻的。

32．给定树叶的权如下：1，4，9，16，25，36，49，64，81，100，试构造一棵最优树。

33．设7个字母在通讯中出现的频率如下，a：35%，b：20%，c：15%，d：10%，e：10%，f：5%，g：5%，编一个相应的二元前缀码，使通讯中出现的字符串尽可能短，画出对应的二元树，对下列信息进行译码。

1 0 0 1 1 0 1 1 1 0 0 0 1 0 1 1 0 1 1 0 0 1 0 1 0 1 0 0 0 1

34．构造一个与英文字母b，d，g，o，y，e对应的前缀码，并画出该前缀码对应的二元树，再用此六个字母构成英文短语，写出此短语的编码信息。

35．设A是二进制序列的集合。现将A划分成两个子集A_0和A_1，这里A_0是A中第一个数字是O的序列的集合，A_1是A中第一个数字是1的序列的集合。然后根据序列中第二个数字将A_0划分为两个子集，对A_1也同样加以划分。运用不断地将序列的集合划分成子集的方法来证明：如果A是前缀码，则存在一棵二元树，其中从每个分枝点引出的边分别标号0和1，使得赋于树叶的0和1的序列是A中的序列。

36．假设有8件产品，它们的外形并无差异，但其中有一件重量不合标准，问怎样以最少次数用天秤找出其中那个不合格产品。

37．现要对一批正数进行分类：凡是不大于20的属于第一类；凡是大于20而不大于50的属于第二类；凡大于50而不大于100的属第三类；其余的属第四类。假定这批数中属第一、二、三、四类数的概率分别是2/18，1/18，5/18，和7/18，试用二元树来描述这一判断过程，并说明二元树具备怎样的形式，过程的执行时间最长或最短。

38．设数组：9，21，5，8，25，7，11，13，17，4。

（1）构造一二元排序树

（2）从树中插入 6，30，24

（3）从树中删除13、25，17，分别画出删除后的树结构。

39．给出下列表达式的有向树表示。

$$(P\vee(\neg P\wedge Q))\wedge((\neg P\vee Q)\wedge\neg R)$$

40．设用“↑”表示乘方，试用有向树表示下列表达式　$(2x+y)(5a-b)^3$

41．对图3.70所示的二元树，试用前序遍历写出访问结果。

42．用中序遍历法重复 41 题。

43．用后序遍历法重复 41 题。

44．对于图3.71，求：

（1）写出图的距离矩阵D。

（2）求出图的中心。

（3）求出图的中位点。

45．求图3.72所示的树的中心和半径。

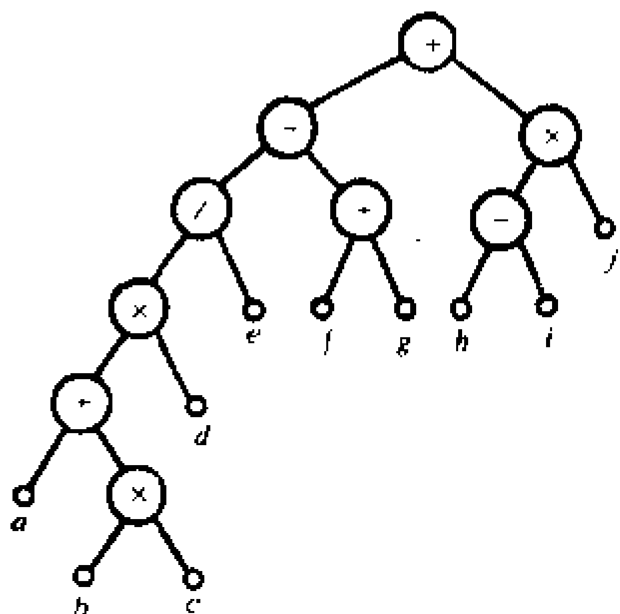


图 3.70

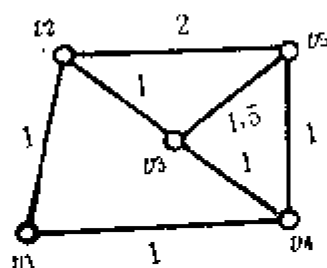


图 3.71

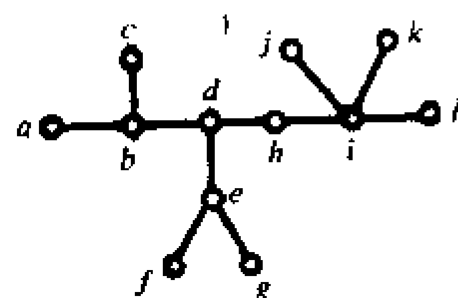


图 3.72

46．设计一程序，求图3.73的连通分支。

47．试用过程 $DFSB$ 求图 3.74 所示的无向图的所有连通块。

48．试用过程 $DFSSCC$ 求图 3.75 所示的有向图的所有强连通块。

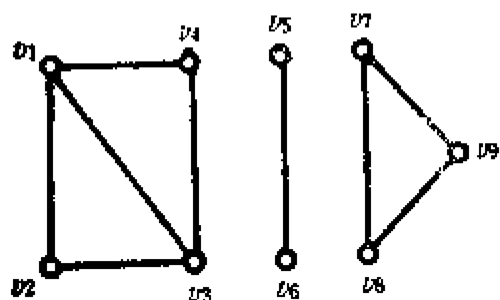


图 3.73

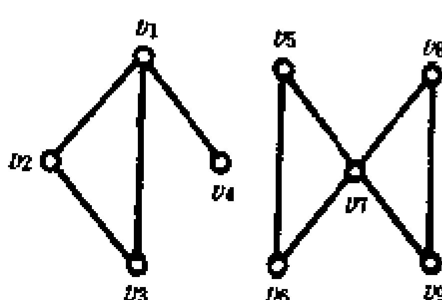


图 3.74

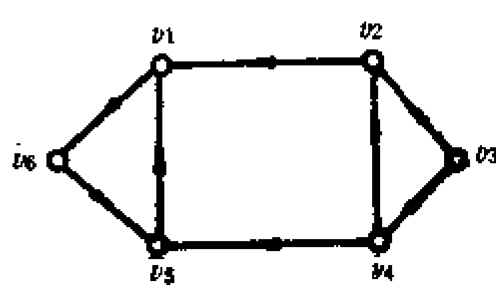


图 3.75

49．试证明：生成树的补不包含割集，而且割集的补不包含生成树。

50．设L是图G的回路，a和b是L中任意两条边，证明存在一个割集K，使$L\cap K=\{a,b\}$。

51．设e是n阶完全图中的一条边，求：

（1）包含e的割集数。

（2）不包含e的割集数。

52．设e_1和e_2是n阶完全图的两条边，求同时包含e_1和e_2的割集数。

53．在K_4的所有四阶子图中，有多少个是割集？有多少个不是割集？

54．证明图G的任一生成树和任一割集至少有一条公共边。

55．证明当且仅当图G的每个结点都处在G的某一回路上时，图G是不可分图。

56．在一连通图G中，设S是具有以下特性的边集合。

（1）S与G的任一割集有偶数条公共边（包括零）。

（2）不存在满足（1）的S的真子集。证明S是一个回路。

57．试以关联集为基底求出图3.76（a），（b），（c）的全部割集。

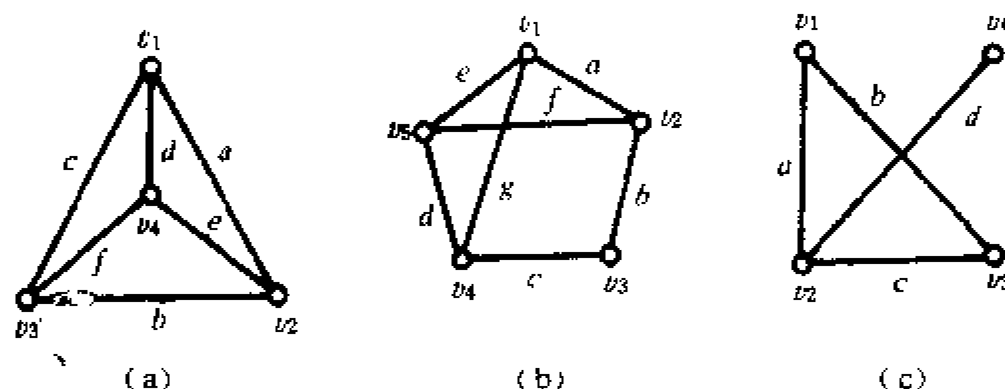


图 3.76

58．对于图3.76（b），任意选定两棵生成树T_1和T_2，分别以它们所对应的基本割集为基底，求出图的全部割集。

59．证明：任意三个不同关联集的交集必为空集。

60．对图3.76（b），选定生成树$T=\{a, b, f, d\}$，写出割集矩阵及基本割集矩阵。

第四章 平 面 图

§4.1 可平面图的概念

在平面上画图时（例如电路图），往往会出现一些边的交叉，称为交叉边。在实际工作中，尽量避免或减少交叉边是很有意义的，例如在制作印刷电路板或交通道路的设计中，都要考虑避免或减少交叉。有些图形表面上看边有交叉，但经过改画以后可以做到边不交叉，如图 4.1(a)的G，表示四座城市A_1，A_2，A_3，A_4及连接它们之间的道路，这时边e_5与e_6出现交叉，如果将图改画成图 4.1(b) 的$\tilde{G}$，则可避免边的交叉。但是有些图无论怎样改画，边的交叉都无法避免，例如图 4.2(a) 的G，A_1，A_2，A_3表示三幢宿舍，B_1,B_2,B_3分别表示抽水站、煤气站及锅炉房，边则表示敷设的水、煤气和暖气管道，理论和实践都证明，这个图无论怎样改画，至少有一条边与其他边交叉，如图 4.2(b)的G'。

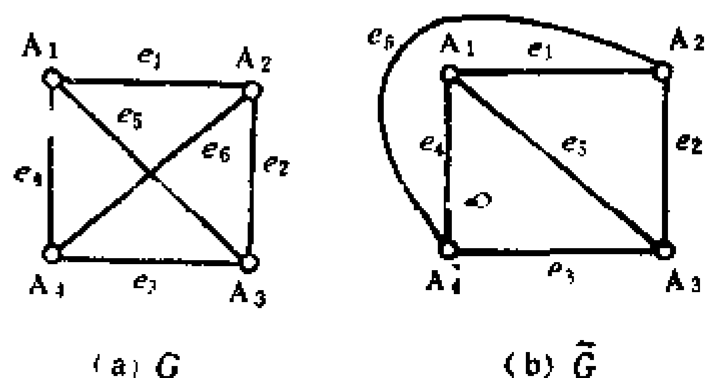


图 4.1

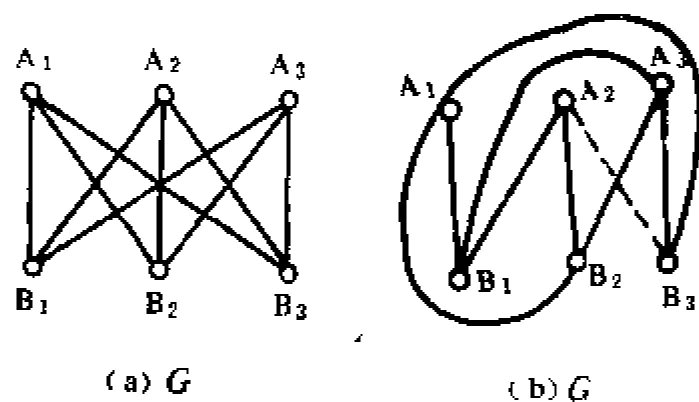


图 4.2

由此提出一个问题：怎样的图画在平面上边不会出现交叉。研究这类图的特征及判别方法，不仅具有理论意义，也有很大的应用价值。本章的内容就是讨论平面图的概念、性质、判别定理，并介绍检测图的平面性的一个算法，在此基础上，介绍图的厚度、对偶图的两种定义及特性，以及 2—同构的概念及判别方法。

定义 4.1 一个图G如果画在平面上能使任意两条边除端点外均不相交，则称G为可平面图，否则称为非可平面图。

一个可平面图G画在平面上，使任意两条边除端点外均不相交，称为将G嵌入平面，G嵌入平面形成的图，称为G的平面表示图。例如图 4.1(a)的G是一个可平面图，(b)中的$\tilde{G}$是G的平面表示图。而图 4.2 的G则是非可平面图。

在第一章中我们已经提到图的拓扑性，一个可平面图与它的平面表示图只是一种拓扑变换，它们是拓扑等价的，以后我们将不再区分而统称之为平面图，并将非可平面图称为非平面图。

定义 4.2 一个平面图G，如果它的一些边所包围的区域内不再有图的结点和边，则称此区域是图G的一个面。

一个面的周界形成的回路常称为网孔，平面图G外部的无限区域也看作是一个面，称为无限面，而网孔所包围的面称为有限面。两个面如至少有一条公共边，则称此两个面是相邻的。

图 4.3 所示的平面图G，有三个有限面R_1，R_2，R_3和一个无限面R_4。

同是一个平面图，不同的画法可以使有限面成为无限面。如图 4.4的（a）可以改画

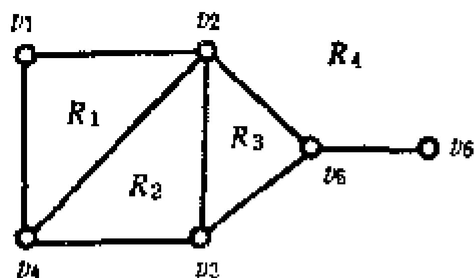


图 4.3

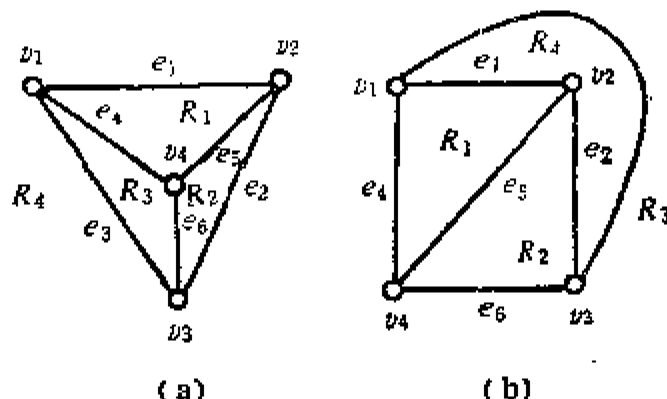


图 4.4

成（b），但在（a）中R_3是有限面R_4是无限面，而在（b）中R_4则成为有限面R_3成为无限面。

可以通过球极平面投影（Stereographic Projection）将平面图的任一有限面变成无限面。所谓球极平面投影就是将平面上的图形投影到球面上或将球面上的图形投影到平面上。方法是将一球S置于平面P上，球与平面的接触点称为球的南极，记作x，球面上端点为x的直径的另一端点记作y称为投影中心（即球的北极）。将平面P上的任一点z与投影中心y用一直线连接，连接线与球面必有一交点z'而且交点是唯一的，于是平面上的所有点与球面上除y点外的所有点建立了一一对应的关系，利用这种投影方法，可将平面上的图形投影到球面上，反之亦然，如图 4.5 所示。

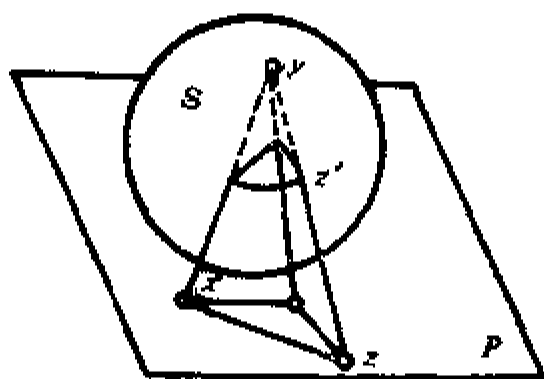


图 4.5

如果要将平面图G的某一有限面R变为无限面，可先将此平面图按球极平面投影法投影到球面上，然后在球面上与R对应的区域内任选一点，转动球使该点成为投影中心y，再把球面上的图投影到平面上，所得到的平面图它的无限面就是R。

由此可见，一个图可嵌入平面上一定也可嵌入球面上，反之亦然。

定义 4.3 在平面图中，构成面R的周界的边的数目，称为面R的次数，记作$deg(R)$。

如图 4.3，$deg(R_1)=3$，$deg(R_2)=3$，$deg(R_3)=3$。这里要指出，如果e不在任一回路中，它必然是无限面的一条边，在计算无限面的次数时，e要计算两次，因此对于图 4.3 的无限面R_4，应有$deg(R_4)=7$。

定理 4.1 一个有限平面图，面的次数和等于边数的二倍，即

$$\sum_{i=1}^{r} deg(R_i)=2m \tag{4.1}$$

其中m为图的边数，r为面数。

证：因为任何一条边，或者是两个面的公共边，或者只是无限面的边，在计算面的周

界时都会重复计算一次，所以面的次数和等于边数的二倍。 ∎

§4.2 平面图的性质

连通平面图的一个重要性质就是它的结点数 n、边数 m 和面数 r（包括无限面）之间存在着一个简单的关系，即下面定理所给出的公式（欧拉公式）

定理 4.2 （欧拉公式） 在一个有 r 个面（包括无限面）的（n，m）连通平面图中，下式成立

$$n - m + r = 2 \tag{4.2}$$

证：用归纳法证明（对面数进行归纳）

设 $r = 1$（一个面），则图不含回路，图又是连通的，故必然是一棵树，根据树的性质，有 $m = n - 1$，因而公式（4.2）成立。

设 $r = 2$，（含两个面），必然一个是有限面，另一个是无限面，即图只有一个网孔，这样的图其形状有如下三种可能

1）图只有一个带自环的结点，如图 4.6(a)

2）图是一个多边形，如图 4.6(b)

3）图只有一个回路，另一些边不构成回路，如图 4.6(c)

所有以上情况，图的结点数与边数的关系均为 $m = n$，因而公式（4.2）亦成立。

设 $r = k$ 时公式成立，即 $n - m + k = 2$，当 $r = k + 1$ 时图的结点数和边数分别以 n' 和 m' 表示，下面证明等式 $n' - m' + (k+1) = 2$ 亦成立。

设面数为 k 时的图为 G，今在 G 的任意两个不相邻接的结点之间加上一条边，得到面数为 $k+1$ 的图 G'，它的边数 $m' = m+1$，结点数 $n' = n$，因而 $n' - m' + (k+1) = n - m + k = 2$，因此公式（4.2）亦成立，定理得证。 ∎

应当指出，欧拉公式只适用于连通平面图，如果图不是连通的，公式不成立，例如图 4.7 是个非连通平面图，它的 $n = 8$，$m = 8$，$r = 3$，则 $n - m + r = 3 \neq 2$。

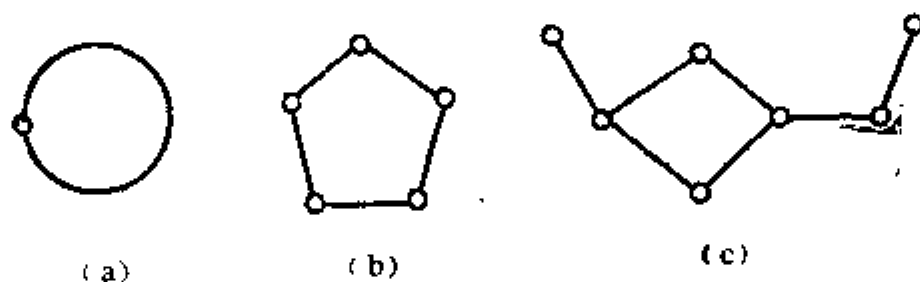


图 4.6

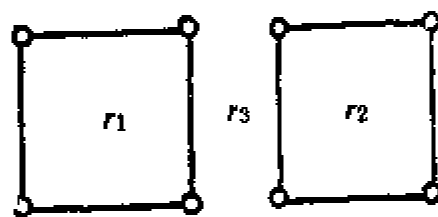


图 4.7

推论 1：设 G 是一个有 n 个结点 m 条边的简单连通平面图，若 $n \geqslant 3$，则

$$m \leqslant 3n - 6 \tag{4.3}$$

证：因 G 是简单平面图，故每个面的边数至少是 3，设有 r 个面，则各面的次数和为

$$\sum_{i=1}^{r} deg(R_i) = 2m \geqslant 3r$$

由式（4.2）得 $r = 2 + m - n$，代入上面不等式即得

$$m \leqslant 3n - 6$$ ∎

推论 2：若G是简单连通平面图，则G至少有一个结点它的次数小于等于5。

证：用反证法，设G的所有结点的次数都大于等于6，则结点次数和

$$\sum_{v\in V} deg(v)=2m\geqslant 6n$$

即$m\geqslant 3n$，与式（4.2）矛盾，故G不可能所有结点的次数都在6以上，即至少有一结点的次数小于等于5。 ▌

推论 3：若G是简单连通平面图且结点数$n\geqslant 3$，则面数

$$r\leqslant 2n-4 \tag{4.4}$$

证：由公式（4.2）和（1.3）得

$$m=r+n-2\leqslant 3n-6$$

故

$$r\leqslant 2n-4$$ ▌

推论 4：若连通平面图(n,m)每个面的次数均为k，则

$$m=\frac{k(n-2)}{k-2} \tag{4.5}$$

证：每一面的次数均为k，则所有面的次数和

$$\sum_{i=1}^{r} deg(R_i)=kr=2m，即 \qquad r=\frac{2m}{k}$$

代入公式（4.2）整理后即得

$$m=\frac{k(n-2)}{k-2}$$ ▌

推论 5：若G是一个不含K_3的简单连通平面图，则

$$m\leqslant 2n-4 \tag{4.6}$$

证：由于不含K_3且是简单连通平面图，故每个面的次数大于等于4，即

$$\sum_{i=1}^{r} deg(R_i)=2m\geqslant 4r，即 \qquad r\leqslant \frac{1}{2}m$$

代入公式（4.2）整理后即得

$$m\leqslant 2n-4$$ ▌

定义 4.4 若在平面图G的任意不相邻两点之间加上一条边，得到的图成为非平面图，则称G为极大平面图。

图4.8所示即是一个极大平面图。

定理 4.3 极大平面图的每个面都由三条边围成。

证：用反证法，设存在一有限面R_i，它由4条以上的边围成，即$deg(R_i)\geqslant 4$，不失一般性，可假定$deg(R_i)=4$，并设其周界形成的回路为$v_1v_2v_3v_4v_1$，如图4.9所示。

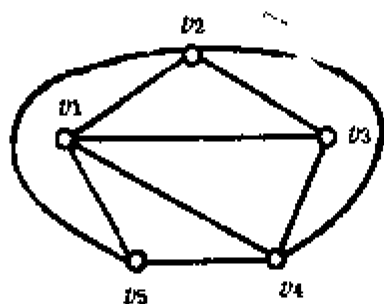


图 4.8

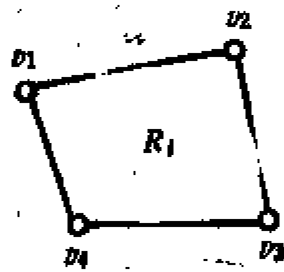


图 4.9

如 v_1 与 v_3 不邻接，则可在面 R_i 内连接 v_1 与 v_3 而不破坏图的平面性，这与 G 是极大平面图相矛盾。如 v_1 与 v_3 已在面 R_i 外连接，则 v_2 与 v_4 在面 R_i 外必无联线（否则联线 v_1v_3 与 v_2v_4 必相交而与平面图的假设相矛盾），这时可在面 R_i 内连接 v_2 与 v_4 而不破坏图的平面性，也与 G 是极大平面图相矛盾，因此有大于等于 4 条边的面的平面图不会是极大平面图，即极大平面图每个面都由三条边围成。 ∎

定理 4.4 极大平面图的结点数 n，边数 m 及面数 r 满足下列等式

$$m=3n-6,\quad r=2n-4 \tag{4.7}$$

证：因极大平面图每个面由三条边围成，所以

$$3r=2m$$

将 $r=\frac{2}{3}m$ 或 $m=\frac{3}{2}r$ 代入公式（4.2）即得

$$m=3n-6,\quad r=2n-4$$ ∎

定理表明当结点数给定时，极大平面图是边数及面数最多的平面图。

定理 4.5 设 G 是结点数 $n\geqslant 4$ 的极大平面图，$\delta(G)$ 为 G 的结点的最小次数，则

$$\delta(G)\geqslant 3 \tag{4.8}$$

证：设 v 是 G 的任一结点，从 G 中去掉 v 及与 v 关联的边得到的图 G' 仍是平面图，结点 v 处在 G' 的一个面内，这个面的周界至少由三条边围成，即周界上至少有三个结点，由于 G 是极大平面图，v 与这个周界上的所有点一定都有边联接，所以结点 v 的次数大于等于 3，命题得证。 ∎

§4.3 平面图的判别

在这一节中我们来探讨怎样判别一个图是平面图还是非平面图。

上一节讲的欧拉定理及其推论所给出的公式，描述了平面图的性质，平面图一定具备这些性质，因此不具备这些性质的图一定不是平面图。

图 4.10 给出的两个图（a）和（b），分别用 K_5 和 $K_{3,3}$ 来表示，对于这两个图，有如下定理。

定理 4.6 K_5 和 $K_{3,3}$ 都是非平面图

证：对于 K_5，$n=5$，$m=10$，则

$$10>3\times 5-6=9$$

不满足公式（4.3），故 K_5 是非平面图。

对于 $K_{3,3}$，图不含 K_3，$n=6$，$m=9$，则

$$9>2\times 6-4=8$$

不满足公式（4.6），故 $K_{3,3}$ 也是非平面图。 ∎

以后称图 K_5 和 $K_{3,3}$ 为基本非平面图。

欧拉公式及其推论，给出了平面图的必要条件而不是充分条件，因此满足这些条件的图不一定是平面图，例如图 $K_{3,3}$ 满足公式（4.3），但它却不是平面图。

1930 年，Kuratowski 给出了判定平面图的一个充要条件，在引出这一定理之前，先介绍图的二次结点内同构的概念。

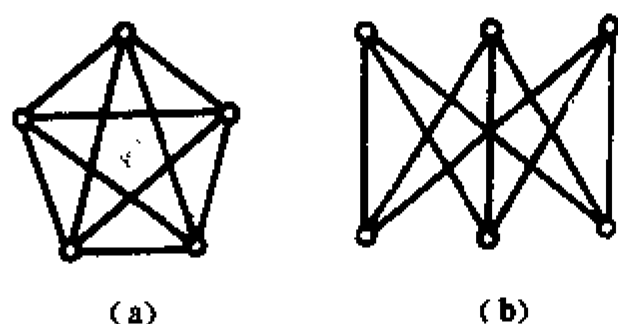


图 4.10

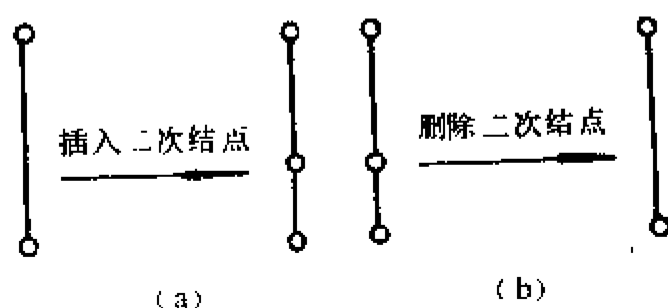


图 4.11

一个图G，如果在它的边上加入一个结点，使一条边分成两条边，我们称为对图G插入二次结点，如图 4.11(a) 所示。若在图G中去掉与两条边关联的结点（称此结点为二次结点）使两条边化成一条边，我们称为对图G删除二次结点，如图 4.11(b) 所示。显然对一个图插入或删除二次结点，都不会影响图的平面性。

定义 4.5 图 G_1 和 G_2 称为是二次结点内同构的，如果它们原来就是同构的，或者通过反复插入或删除二次结点后而成为同构的。

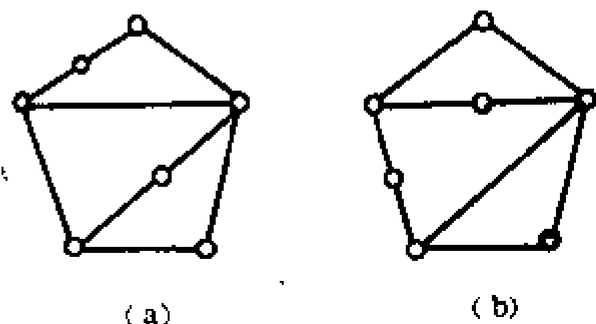


图 4.12

图 4.12 的(a),(b)两个图是二次结点内同构的。由图可见，尽管二次结点内同构的两个图有着不同的结点数和边数，但它们具有相同的面，所以从图的平面性来看，二次结点内同构的图具有相同的特征。

定义 4.6 设 $G_1=(V_1, E_1)$ 是图 $G=(V, E)$ 的一个子图，称下述边的集合为图G相对 G_1 的片（Piece）。

（1）一条边 $(u, v)\in E$ 且 $u, v\in V_1$ 但 $(u, v)\notin E_1$。

（2）$G-G_1$ 中的连通分支，其中任意两点之间的简单路径，除端点外不含 G_1 的结点。

片与 G_1 的公共结点称为片的接触点。具有两个或更多接触点的片称为桥（Bridge）。

例 4.1 如图 4.13(a) 所示的图G，取回路 $C=(v_1, v_2, v_3, v_4, v_5, v_1)$ 作为子图 G_1（图中的粗线），则G相对 G_1 的片有四个，如图 4.13(b)，它们分别是

$$B_1=\{(v_3, v_5)\}$$

B_1 的接触点是 v_3，v_5。

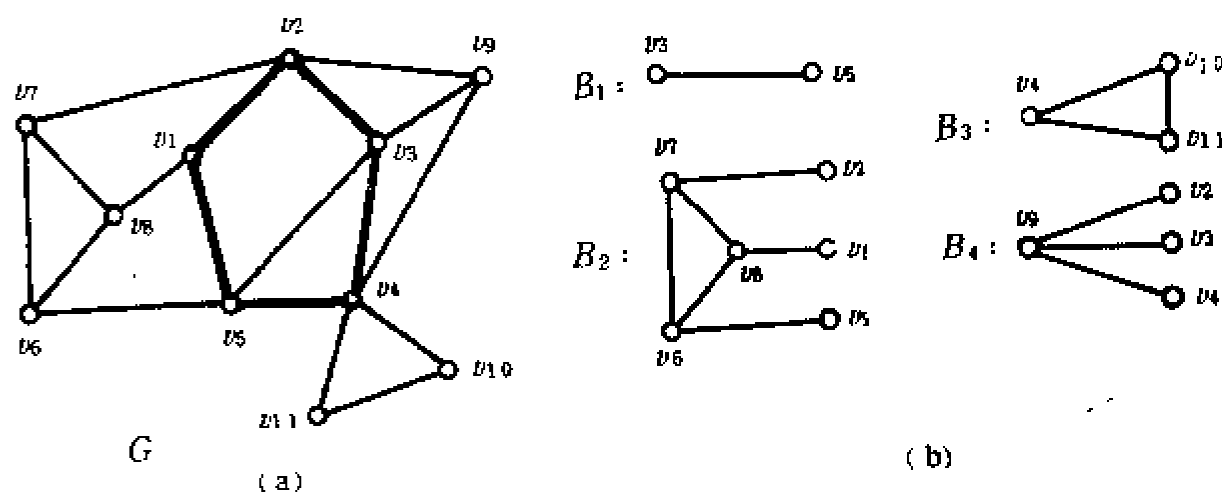


图 4.13

$$B_2=\{(v_2,v_7),(v_1,v_8),(v_5,v_8),(v_6,v_7),(v_7,v_8),(v_6,v_8)\}$$

B_2 的接触点是 v_2, v_1, v_5。

$$B_3=\{(v_4,v_{10}),(v_4,v_{11}),(v_{10},v_{11})\}$$

B_3 的接触点是 v_4。

$$B_4=\{(v_2,v_9),(v_3,v_9),(v_4,v_9)\}$$

B_4 的接触点是 v_2,v_3,v_4。

B_1,B_2,B_4 都有两个以上的接触点，因而都是桥，B_3 则不是桥。

定理 4.7 （**Kuratowski**） 一个图是平面图当且仅当它不含与 K_5 或 $K_{3,3}$ 二次结点内同构的子图。

证：必要性：设图 G 是平面图，显然它不可能含有与 K_5 或 $K_{3,3}$ 二次结点内同构的子图，否则这类子图不可能嵌入平面，整个图 G 也就不可能嵌入平面，与前提条件矛盾，必要性得证。

充分性：（这一部分证明较长，读者可以略去不读。）对边数用归纳法证明如下。

显然只有一条边或两条边的图一定是平面图。设少于 m 条边的图命题成立，我们现在证明具有 m 条边的图命题亦成立。用反证法，即设 G 不是平面图，但不含与 K_5 或 $K_{3,3}$ 二次结点内同构的子图。通过证明将导致矛盾。

如果 G 不是平面图，下述推论成立。

（1） G 一定是连通图。否则 G 的各个连通分支的边都将少于 m，根据归纳假设都是平面图，因而 G 也将是平面图。

（2） G 不含割点。否则 G 可以被割点 x 分割开，由（1）知每一分支都是平面图，对每一分支通过球极平面投影，可将割点 x 投影到平面图的无限面上，于是各分支通过 x 连接后仍将保持其平面性，即 G 将是平面图。

（3） 去掉 G 的任意一条边 $e=(x,y)$，得到图 $G'=G-(x,y)$，则图 G' 中必有穿过 x 和 y 的简单回路。这是因为 G 不含割点，所以 G' 是连通的，如果 G' 不存在这样的简单回路，那么从 x 到 y 的任何一条路径都必将通过一公共结点，比如说 z，如图 4.14 所示，则 z 将是 G' 的割点，z 将 G' 分割为两个连通分支 B_1 和 B_2，B_1 和 B_2 的边数都少于 m，故 B_1 和 B_2 应是平面图，x 和 y 分别处于 B_1 和 B_2 之中，通过球极平面投影可以将 x 与 y 同时投影到平面图的无限面上，然后连接 x 和 y 恢复成图 G，G 将是平面图，与前提矛盾。

综上所述，图 $G'=G-(x,y)$ 是 G 的连通子图，比 G 少一条边，因而根据归纳假设 G' 是平面图，并有一条穿过结点 x 和 y 的回路 C，事实上这样的回路可能不止一条，我们选择其中最大的一条回路作为 C，所谓 C 最大，就是 C 的内部面包含 G' 尽可能多的边，或者说不能再经过球极平面投影将 C 的外部面的图嵌入 C 的内部面而不失去平面性。

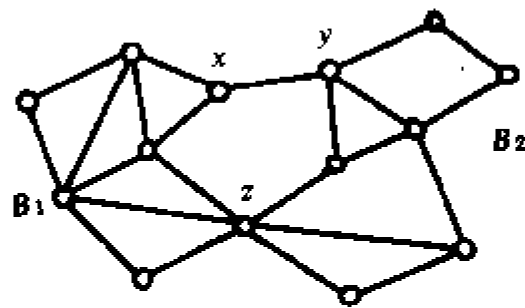


图 4.14

为了叙述方便，我们设定 C 的一个方向比如取顺时针方向，如果 a 和 b 是 C 上的结点，用 $S[a,b]$ 表示顺时针沿着 C 从 a 到 b （含 a 和 b） 的点的集合，并用 $S(a,b]$, $S[a,b)$, $S(a,b)$ 分别表示 $S[a,b]-\{a\}$, $S[a,b]-\{b\}$, $S[a,b]-\{a,b\}$。

由于G是从平面图G'上加一条边（x,y）而构成的，因此G'相对C至少有一外部片及一内部片，否则连接边（x,y）构成的G将是平面图，与题设矛盾。下面我们就来分析相对C的外部片和内部片的情况。

因为G'是连通的，C的外部片必然与C连接，又因C是最大的，每个外部片与C至少有两个接触点（否则这个外部片可嵌入C内，与C是最大的相矛盾）而且至多有两个接触点，这是因为如果接触点多于两个，则将存在穿过结点x，y且内部边的数目比C更多的回路C'，如图4.15所示（图中粗线表示回路C'），这与C的选择矛盾，由此可知，C的所有外部片都是与C恰有两个接触点的桥，称为外部桥，它的两个接触点设为i，j。不仅如此，而且如果$i\in S(x,y)$，则必有$j\in S(y,x)$，这是因为如果i，j都在x，y的同一侧，如图4.16，也将存在比C更大的回路C'（图中粗线表示C'）。因此，任一外部桥，它的接触点分布都可用图4.17表示。

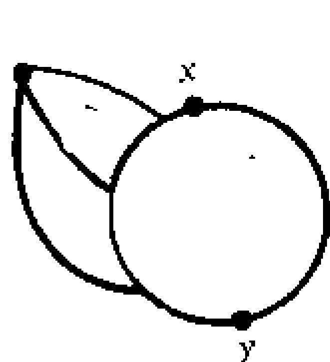


图 4.15

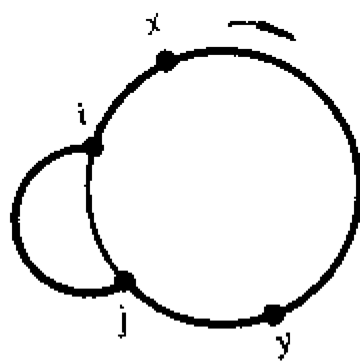


图 4.16

对于内部片，至少有一片与C有两个接触点，设为a和b，而且如果$a\in S(x,y)$，则必有$b\in S(y,x)$，否则便可将边（x,y）加上，恢复成图G，G将是平面图，导致矛盾。同时至少有一内部片，它的两个接触点与某一外部片的两个接触点是交叉的，否则这一外部片就可以嵌入C的内部而不会破坏其平面性，与C是最大回路相矛盾。图4.17表示了内部片的两个接触点恰好同时满足这两个条件的情况。

由于内部片的接触点可能不止两个，下面根据它们的不同分布情况分别进行讨论。这里我们是将边（x,y）加到平面图G'上，使它恢复成图G，对G进行讨论的。

情况 1 接触点a，b与x，y，i，j均不重合且形成相互交叉如图4.18(a)所示，则G包含一个与$K_{3,3}$二次结点内同构的子图，如图4.18(b)，图中我们把与$K_{3,3}$对应的两个结点子集的结点分别用空心圆点和实心圆点表示。

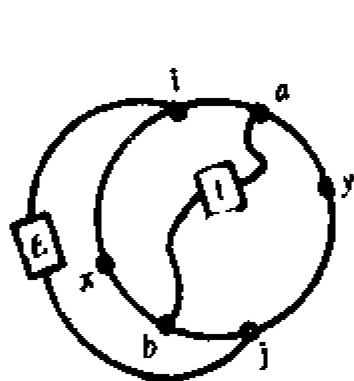
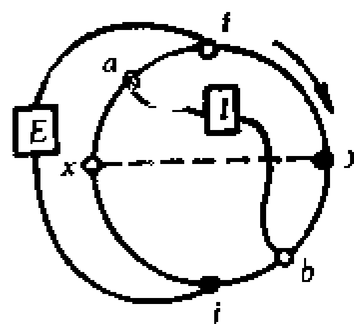
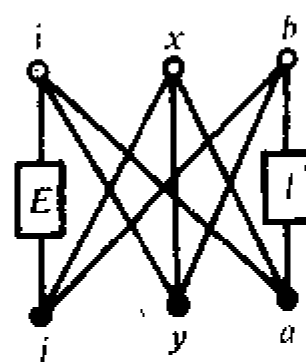


图 4.17



(a)



(b)

图 4.18

情况 2　a，b 在 x，y 的同一侧，不妨设 a，$b \in S(x, y)$，此时必须还有一个接触点 $c \in S(y, x)$，并且内部片上必存在一结点 v_0 以及分别连接 a, b, c 的三条通路。根据 c 点的位置，有如下三种可能：

（1） $c \in S(y, j)$，如图 4.19(a)，则 G 包含一个与 $K_{3,3}$ 二次结点内同构的子图。（与情况 1 相同，对应的结点集为 $\{j, y, a\}$ 与 $\{i, x, c\}$）

（2） $c \in S(j, x)$，如图 4.19(b)，则 G 也包含一个与 $K_{3,3}$ 二次结点内同构的子图。（与情况 1 同，对应的结点集为 $\{i, y, c\}$ 与 $\{j, x, b\}$）

（3） $c = j$，如图 4.19(c)，则 G 也包含一个与 $K_{3,3}$ 二次结点内同构的子图。（对应的结点集为 $\{i, x, v_0\}$ 与 $\{j, b, a\}$）

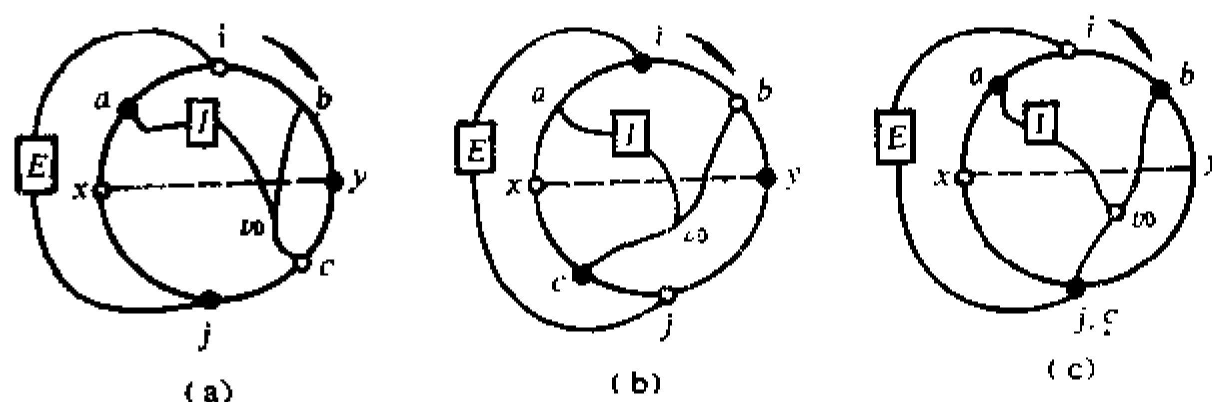


图 4.19

与此类似，当 a，$b \in S(y, x)$ 时，也将得出与上述相同的结果。

情况 3　$a = x$，$b \in S(x, y)$，此时必须还有一个接触点 $c \in S(y, x)$，而且内部片上必存在一结点 v_0 以及由 v_0 连接 a, b, c 的三条通路。c 的位置亦可能有三种即：$c \in S(y, j)$，$c \in S(j, x)$ 及 $c = j$，分别如图 4.20(a)，(b)，(c) 所示。可见无论哪种情况，G 都包含一个与 $K_{3,3}$ 二次结点内同构的子图。

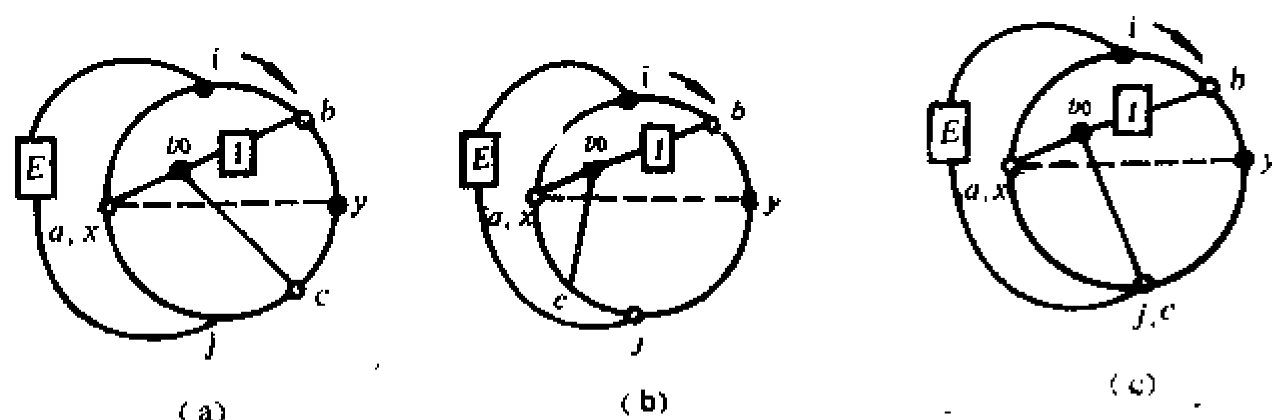


图 4.20

情况 4　$a = x$ 且 $b = y$，此时内部片必须还有两个接触点 c 和 d，且 $c \in S(x, y)$，$d \in S(y, x)$。如果连接 a 与 b 的通路和连接 c 与 d 的通路有一个以上的公共结点，则图 G 包含一个与 $K_{3,3}$ 二次结点内同构的子图，如图 4.21 所示，图中(a)，(b)，(c)，(d) 分别表示接触点 c，d 处子不同位置的情况。

如果连接 a 与 b 的通路和连接 c 与 d 的通路只有一个公共结点 v_0，如图 4.22 的(a)，(b)，(c)，图 G 都包含一个与 $K_{3,3}$ 二次结点内同构的子图。只有当 $c = i$ 及 $d = j$ 时，图 G 包含一个与 K_5 二次结点内同构的子图，如图中的(d)所示。

以上所有可能情况都证明，如 G 是非平面图，则必定包含与 $K_{3,3}$ 或 K_5 二次结点内同构的子图，所以不含与 $K_{3,3}$ 或 K_5 二次结点内同构的子图的图，一定是平面图。充分性

得证。 ▮

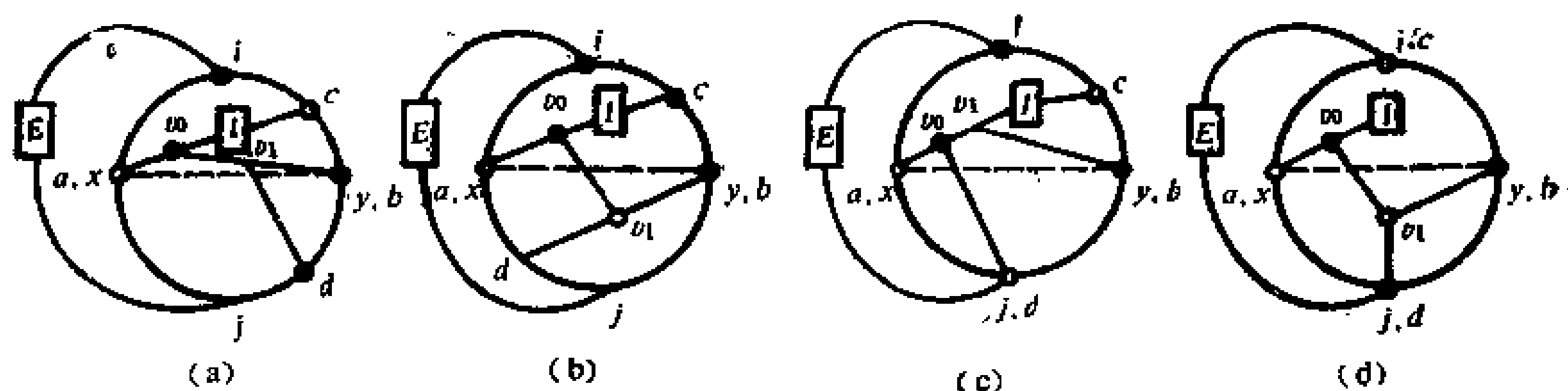


图 4.21

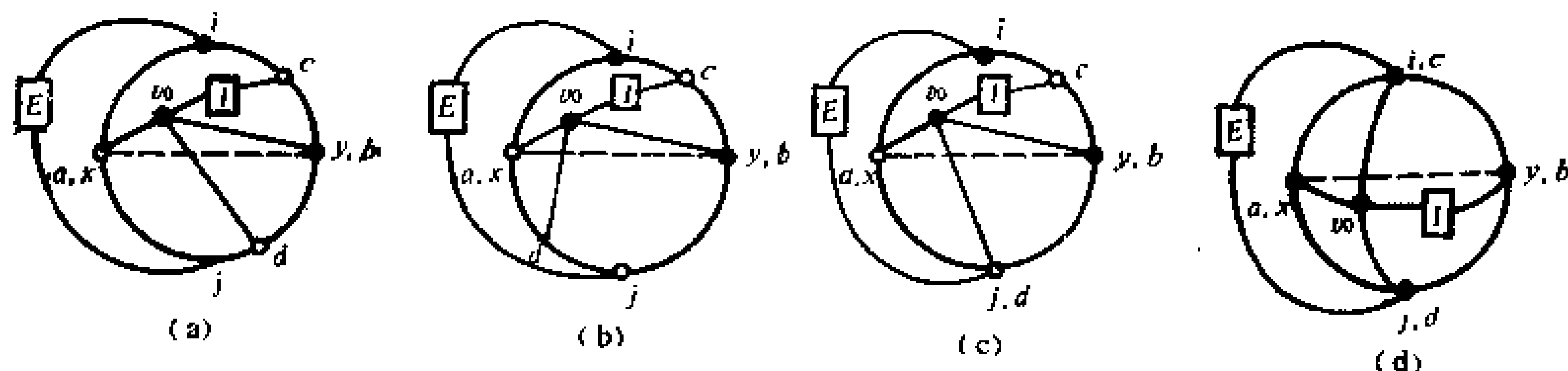


图 4.22

例 4.2 如图 4.23(a)称为彼特森(Petersen)图，去掉边 (v_3, v_4) 和 (v_7, v_{10}) 得到子图 G'，如图中(b)或(c)所示，可见G'与$K_{3,3}$二次结点内同构，因此彼特森图是非平面图。

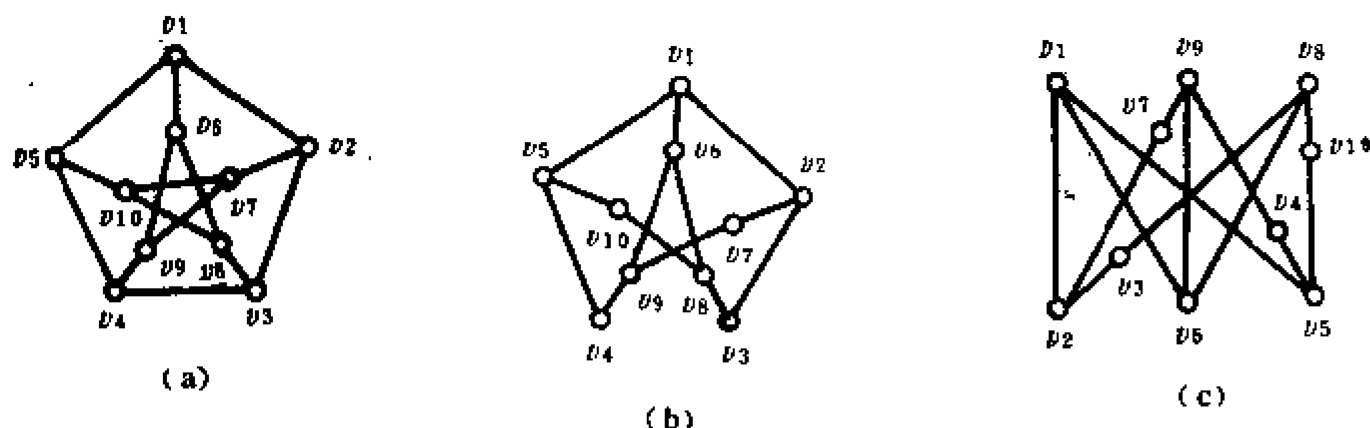


图 4.23

§4.4 平面性算法

上一节我们介绍了怎样判别一个图是否是平面图的方法，即库拉托夫斯基定理，从理论上讲，这个定理简单明确，园满地解决了图的平面性判别问题，但实际上要判定一个图是否存在与$K_{3,3}$或K_5二次结点内同构的子图并非易事，从算法的角度讲这个判别方法是很复杂的。因此寻求一个简单有效的算法，就成为检测图的平面性的重要课题。这一节我们将介绍一个较好的算法，并给出一个例子。

通常应用算法对一个给定的图进行平面性检测时，都要对图施行一些简化处理，使问题更为简单，简化处理的步骤如下。

（1）如果图G不是连通图，则需要对它的每一个连通分支进行检测

（2）如果图是可分图（即含有割点)，显然图是平面图当且仅当它的每一个连通块都是平面图，因此可以对每个连通块分别进行检测。

（3）因为增加或删除自环不会影响图的平面性，可以将图的所有自环删除。

（4）平行边也不影响图的平面性，在每一对结点之间，除保留一条边外，删除其余的平行边。

（5）二次结点不影响图的平面性，与二次结点关联的两条串联边可以用一条边来代替（即删除二次结点）。

反复进行第（4），（5）步直到不能再简化为止，一般将能大大地简化一个图。这时可进行如下简单测试：

（1）如果$m<9$或$n<5$则G一定是平面图。（m，n分别表示G的边数和结点数。）

（2）如果$m>3n-6$，则G一定是非平面图。

如果上述两条均不满足，则可用平面性算法进行检测。

例4.3 用上述步骤对图4.24(a)进行简化，可见图G是平面图。

下面介绍由 Demoucron, Malgrange 和 Pertuiset 于 1964 年提出检测平面性的一个算法。先引入一个定义如下。

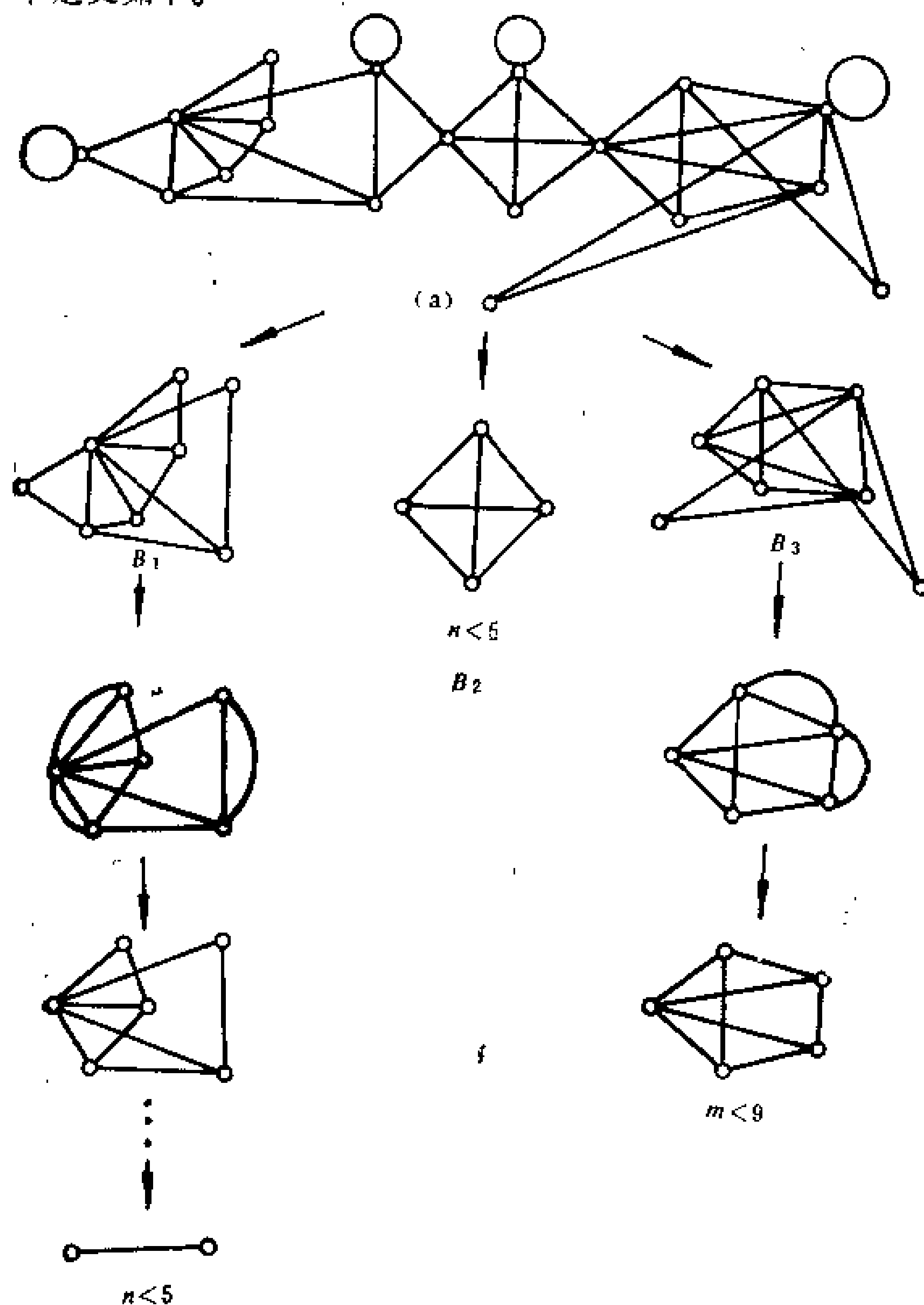


图 4.24

定义4.7 设H是图G的一个可平面子图，$\tilde{H}$是H的一个平面表示图，B是G相对H的任一个桥，如果B与H的接触点全都在$\tilde{H}$的同一个面的边界上，则称B是可画入的，并以$F(B,\tilde{H})$表示在$\tilde{H}$中所有含B的全部接触点的面的集合，如果没有一个面含B的全部接触点，则$F(B,\tilde{H})=\phi$

例4.4 如图4.25,(a)为图G,(b)是G的一个可平面子图H,(c)表示H的一个平面表示图$\tilde{H}$及桥B_1和B_2（桥用粗线表示），可见

$$F(B_1,\tilde{H})=\{R_0,R_1\}$$

$$F(B_2,\tilde{H})=\phi$$

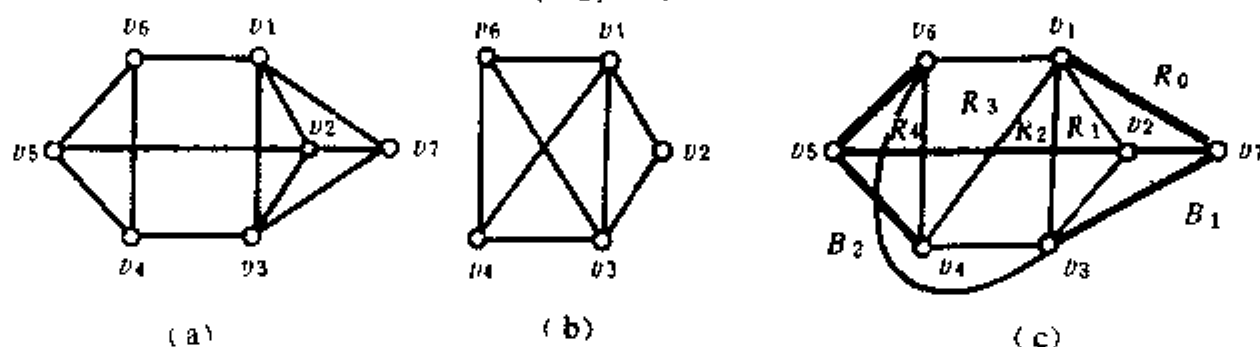


图 4.25

*D.M.P*平面性算法

1. 找出G的一个回路C
2. $i\leftarrow 1$
3. $G_1\leftarrow C$, $\tilde{G}_1\leftarrow C$
4. $r\leftarrow 2$
5. EMB←true
6. While $r\neq m-n+2$ and EMB do

   begin

7. 求出G相对G_i的每个桥B
8. 对每一个桥B，求$F(B,\tilde{G}_i)$
9. if 存在桥B, $F(B,\tilde{G}_i)=\phi$ then

   begin

10. EMB←false
11. 输出“G是非平面图”

    end

12. if EMB then

    begin

13. if 存在桥B, $|F(B,\tilde{G}_i)|=1$ then $R\leftarrow F(B,\tilde{G}_i)$else任取一$B$和$R$使$R\in F(B,\tilde{G}_i)$
14. 找一通路$P_i\subseteq B$, 使它连接B相对G_i的两个接触点
15. $G_{i+1}\leftarrow G_i+P_i$
16. 将P_i画入$\tilde{G}_i$的面R中，得到G_{i+1}的平面表示图$\tilde{G}_{i+1}$
17. $i\leftarrow i+1$
18. $r\leftarrow r+1$
19. if $r=m-n+2$ then 输出“G是平面图”

    end

end

算法说明：

算法是通过求图的序列 $G_1, G_2, \cdots (G_i \subset G_{i+1})$ 以及它们的平面表示图 $\tilde{G}_1, \tilde{G}_2 \cdots$ 来判定图 G 的平面性的。如果 G 是平面图，我们将得到算法求出的每一 $\tilde{G}_i$ 都是 G 的平面子图，当求出 $\tilde{G}_{m-n+1}$ 时算法结束，得到的即是 G 的平面表示图。如果 G 是非平面图，则算法在发现存在桥 B（相对当前求出的 G_i）有 $F(B, \tilde{G}_i)=\phi$ 时即停止，所以相对 G_i 的每一个桥 B，$F(B, \tilde{G}_i)\neq\phi$ 是 $\tilde{G}_i$ 可成为 G 的平面子图的必要条件。由算法求图的序列的第一个图 G_1 是一个回路（算法1—3行），前面已讲过在执行这一算法前已对图进行予处理（简化）。所以这时图都是不含割点的连通块，因此一定能够找到图的一个回路，显然 G_1 是平面图。算法设置了一个布尔变量 EMB，对当前的 $\tilde{G}_i$，只要没有发现 B 有 $F(B, \tilde{G}_i)=\phi$，则EMB 均为真（5，6，12 行），否则 EMB 为假，算法结束并输出信息"G 是非平面图"。（9，10，11 行）。变量 r 用作记录当前 $\tilde{G}_i$ 面的数目，其初始值为 2，每执行一次循环体 While（7—19 行）其值加 1。并从 $\tilde{G}_i$ 构造出一个新的图 $\tilde{G}_{i+1}$，过程如下，先求出 G 相对 G_i 的全部桥 B 以及每个桥的 $F(B, \tilde{G}_i)$，（7，8 行），如果有 $|F(B, \tilde{G}_i)|=1$（13行），说明这个桥 B 能画入 $\tilde{G}_i$ 的唯一的一个面 R 中，这时找出 B 与面 R 的两个接触点之间 B 中的一条路径 P_i，将 P_i 画入 $\tilde{G}_i$ 的面 R 内即得 $\tilde{G}_{i+1}$。如果不存在具备这个条件的桥，则可在任意一个桥 B 中找这样一条路径 P_i，将它画入 $F(B, \tilde{G}_i)$ 的任一面中即得 $\tilde{G}_{i+1}$，（14—16 行）。上述过程，P_i 都将使某个面 R 划分为两个面，于是 r 的值加 1（18 行），如果 G 是平面图，由欧拉公式，面数 r 等于 $m-n+2$，算法结束并输出信息"G 是平面图"。如果对算法稍作补充，使计算 $\tilde{G}_i$ 时用它的面的集合 $\{R_i\}$ 来表示，每个面又用它周界上的结点按一定方向（比如顺时针方向）的序列来表示，则最后输出的平面图可用它的面的集合表示。

可以看出，算法的时间复杂性主要在于执行循环体 While 所耗费的时间，这一循环体至多执行 $m-n+2$ 次，它是以多项式函数为界的，虽然高于线性时间函数，但仍不失为一较好的算法。

例 4.5 用 D. M. P 算法判断图 G（图 4.26）的平面性。

解：一、求 $\tilde{G}_1$

任取 G 的一个回路

$$C=(v_1, v_2, v_3, v_4, v_5, v_6, v_1)$$

$$\tilde{G}_1=G_1=C$$

如图 4.27，于是相对 G_1 的桥有

$B_1=\{(v_2, v_7)\}$

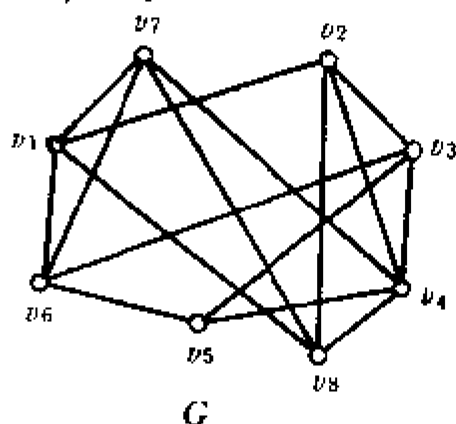


图 4.26

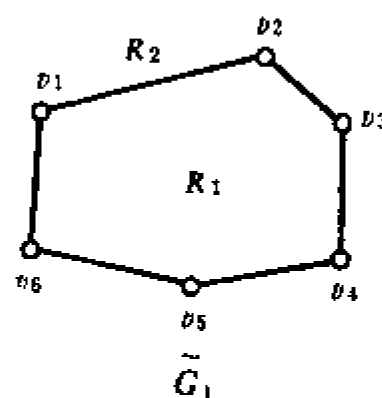


图 4.27

$B_2=\{(v_3, v_5)\}$

$B_3=\{(v_3, v_6)\}$

$B_4=\{(v_7, v_1), (v_7, v_6), (v_7, v_4), (v_7, v_8), (v_8, v_2), (v_8, v_1), (v_8, v_4)\}$

$$F(B_1, \tilde{G}_1)=F(B_2, \tilde{G}_1)=F(B_3, \tilde{G}_1)=F(B_4, \tilde{G}_1)=\{R_1, R_2\}$$

二、求$\tilde{G}_2$

选择桥B_1及面R_1，因B_1只有两个接触点v_2和v_4，且连接v_2和v_4的只有一条边(v_2, v_4)，故取路径$P_1=v_2v_4$，将P_1画入$\tilde{G}_1$的面R_1中使R_1分为两个面R_3和R_4，于是得到图$\tilde{G}_2$如图4.28所示。此时

$$F(B_2, \tilde{G}_2)=F(B_3, \tilde{G}_2)=\{R_2\}$$

$$F(B_4, \tilde{G}_2)=\{R_2, R_4\}$$

三、求$\tilde{G}_3$

取桥B_2及面R_2，得$P_2=v_3v_5$，将P_2画入$\tilde{G}_2$的面R_2中使R_2分为两个面R_5和R_6，于是得到$\tilde{G}_3$，如图4.29，此时

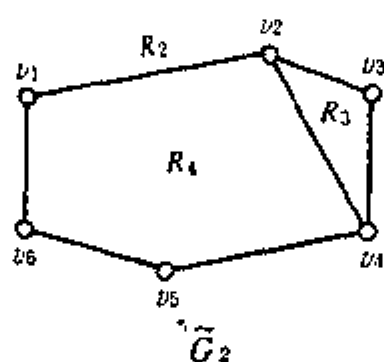


图 4.28

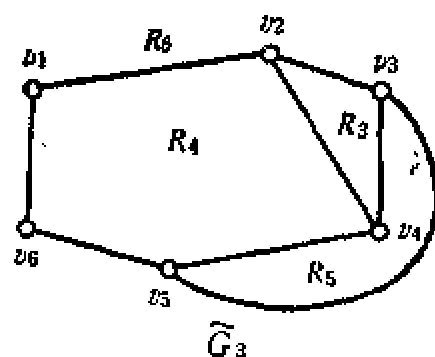


图 4.29

$$F(B_3, \tilde{G}_3)=\{R_6\}$$

$$F(B_4, \tilde{G}_3)=\{R_4\}$$

四、求$\tilde{G}_4$, $\tilde{G}_5$

取桥B_3及面R_6将$P_3=v_3v_6$画入$\tilde{G}_3$的面R_6中将R_6分为R_7和R_8并得到$\tilde{G}_4$如图4.30。接着取桥B_4及面R_4，由于B_4与R_4周界的接触点有v_1, v_4, v_6，任取其中两点比如v_1与v_4之间的一条路径$P_4=v_1v_7v_4$，将它画入$\tilde{G}_4$的面R_4中，R_4被分为R_9和R_{10}并得到$\tilde{G}_5$，如图4.31。

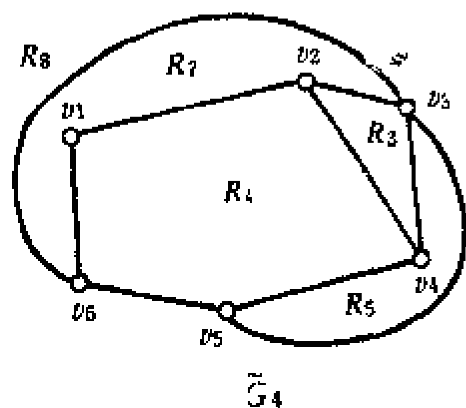


图 4.30

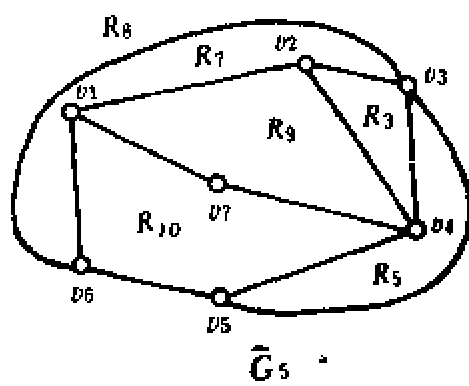


图 4.31

相对于$\tilde{G}_5$有桥

$$B_5=\{(v_6, v_7)\}$$

$$B_6=\{(v_8, v_1), (v_8, v_7), (v_8, v_2), (v_8, v_4)$$

$$F(B_5, \tilde{G}_5)=\{R_{10}\}$$

$$F(B_6, \tilde{G}_5)=\{R_9\}$$

五、求 $\tilde{G}_6$, $\tilde{G}_7$

取桥 B_5 及面 R_{10}, 将 $P_5=v_6v_7$ 画入 $\tilde{G}_5$ 的面 R_{10} 中得 $\tilde{G}_6$, 如图4.32。接着取桥 B_6 及面 R_9, 在 B_6 中取 v_1 与v_4 之间的路径 $P_6=v_1v_8v_4$ 画入 $\tilde{G}_6$ 的面 R_9 中得 $\tilde{G}_7$, 如图4.33。相对于 $\tilde{G}_7$ 有

$$B_7=\{(v_8, v_2)\}, B_8=\{(v_8, v_7)\}$$

$$F(B_7, \tilde{G}_7)=\{R_{14}\}, F(B_8, \tilde{G}_7)=\{R_{13}\}$$

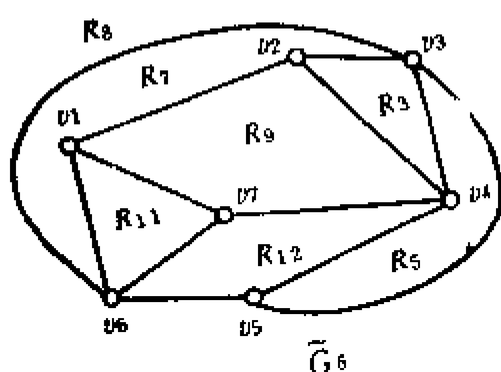


图 4.32

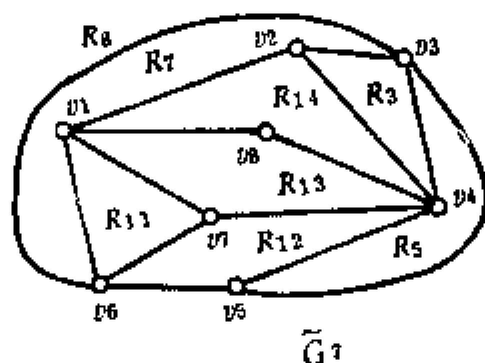


图 4.33

六、求 $\tilde{G}_8$, $\tilde{G}_9$

取桥 B_7 将 $P_7=v_8v_2$ 画入 $\tilde{G}_7$ 的面 R_{14} 中得 $\tilde{G}_8$, 如图4.34。接着将 B_8 的 $P_8=v_8v_7$ 画入 $\tilde{G}_8$ 中得 $\tilde{G}_9$, 如图4.35。此时有 $r=m-n+2=10$, 程序结束，所得到的 $\tilde{G}_9$ 即为G的一个平面表示图 $\tilde{G}$。

最后要说明一点，在执行这一算法中，当求 $\tilde{G}_1$ 时如果取不同的回路C，或者在执行算法第13行时取不同的B和R，则得到G的平面表示图 $\tilde{G}$ 亦将不同，但它们都是拓朴等价的，即可通过球极平面投影相互变换。

例4.6 用D.M.P算法判断图 4.36 中图G的平面性。

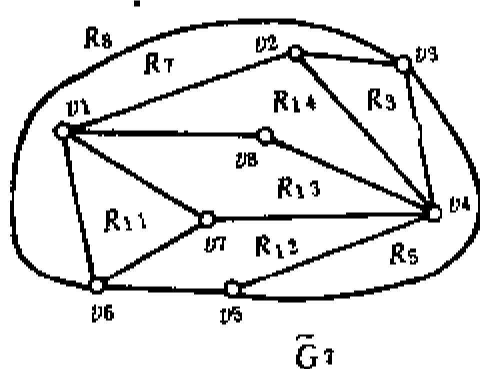


图 4.34

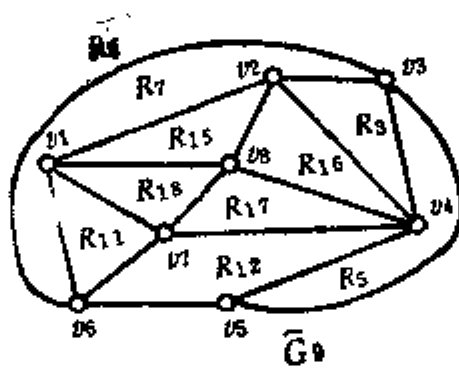


图4.35

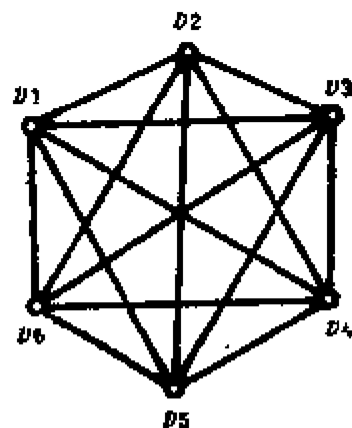


图 4.36

解: 取G的一个回路

$$C=(v_1, v_3, v_5, v_1)$$

得 $\tilde{G}_1$ 及相对 $\tilde{G}_1$ 的桥 B_1, 如图 4.37 所示，此时 $F(B_1, \tilde{G}_1)=\{R_1, R_2\}$取路径

$$P_1=v_1v_6v_4v_2v_3$$

将它画入 $\tilde{G}_1$ 的面 R_2 中得 $\tilde{G}_2$ 及相对 $\tilde{G}_2$ 的 B_2, 如图 4.38, 此时 $F(B_2, \tilde{G}_2)=\{R_4\}$，取路径

$$P_2 = v_5 v_4$$

将它画入 $\tilde{G}_2$ 的面 R_4 中得 $\tilde{G}_3$ 及相对 $\tilde{G}_3$ 的B_3，如图 4.39，此时

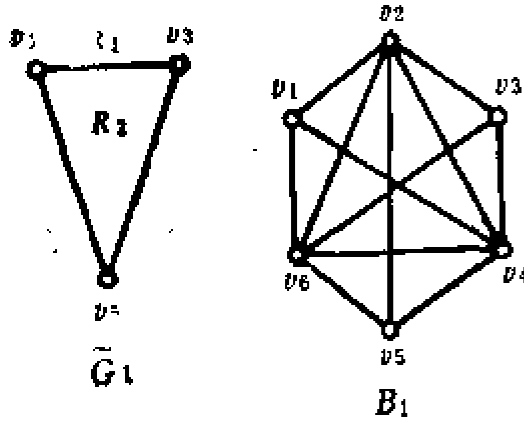


图 4.37

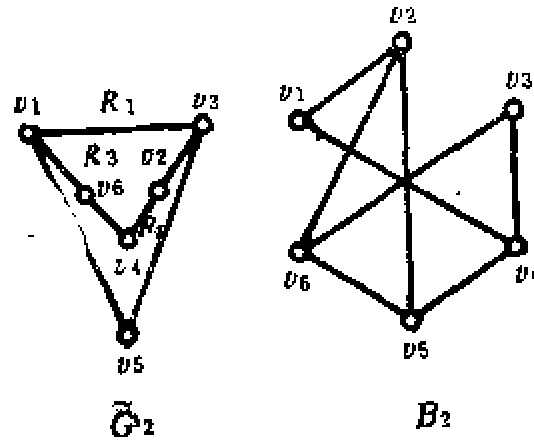


图 4.38

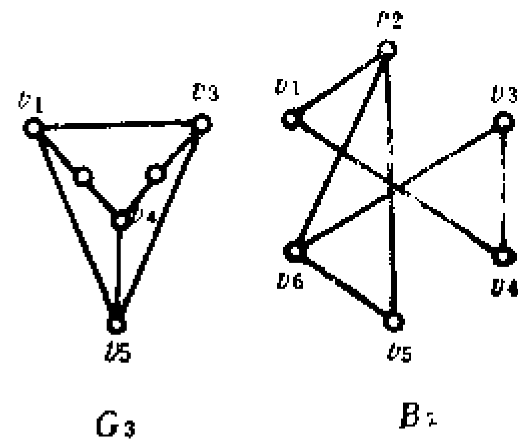


图 4.39

$$F(B_3, \tilde{G}_3) = \phi$$

结果得出G是非平面图，算法结束。

§4.5 图的交叉和厚度

把一个非平面图画在平面上，不可避免地会出现边的交叉，我们把必不可少的交叉数称为图的交叉数。例如两个基本非平面图 $K_{3,3}$ 和 K_5 的交叉数都是 1，而平面图的交叉数则为零。对于一般的非平面图的交叉数，目前还没有得出一个普遍适用的计算公式。

有些问题，例如印刷电路、大规模集成电路的设计中，是不允许出现边的交叉的。为了避免交叉，一种方法是将交叉边通过电路变换成为不交叉，如图 4.40(a)的 (X, X') 与 (Y, Y')是两条交叉边，如果在电路中加入三个或一非门，如图中的(b)，则可避免交叉。

避免交叉的另一种方法，可将一个非平面图分成若干个平面子图，每个平面子图分别画在不同的平面上。例如设计的一个印刷电路，如果不能在一块印刷电路板上实现，可将它分成若干块印刷电路板，那么，至少需要多少块印刷电路板呢？这个问题无论从工艺还是从经济上考虑，都是很有意义的。

定义 4.8 将一个非平面图G分为若干个子图，使每个子图都是平面图，所需子图的最少数目，称为图G的厚度，记作 $\theta(G)$。

由定义可知，平面图的厚度为 1，基本非平面图 K_5 和 $K_{3,3}$ 的厚度都是 2。图4.41 表示完全图 K_9，可见

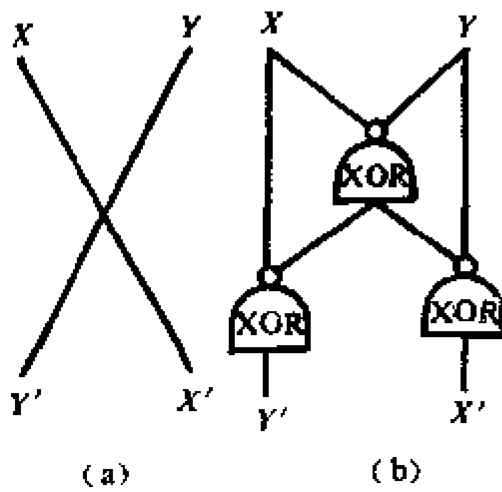


图 4.40

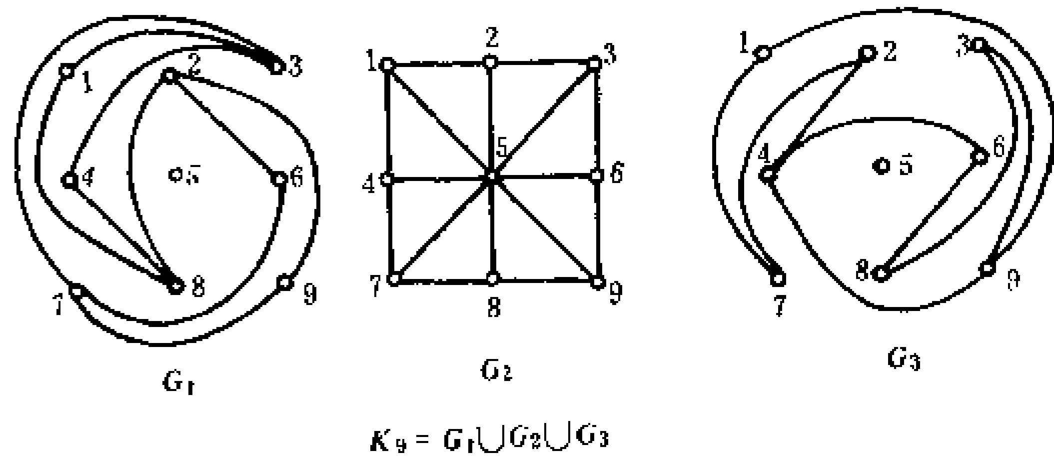


图 4.41

$$\theta(K_9)=3$$

K_9 在第一和第二个平面上的边都已构成了平面图，剩下的边画在第三个平面上，因此它的厚度是 3 。

定理4.8 非平面图G的厚度

$$\theta(G)\geqslant\left\lceil\frac{m}{3n-6}\right\rceil \tag{4.9}$$

若G不含 K_3, 则有

$$\theta(G)\geqslant\left\lceil\frac{m}{2n-4}\right\rceil \tag{4.10}$$

式中m，n分别为图的边数和结点数。$\lceil x\rceil$表示大于等于x的最小整数。

证：根据式（4.3），对平面图，必有

$$m\leqslant 3n-6$$

不满足上式的图一定是非平面图，因此对非平面图，一般情况下有

$$m\geqslant 3n-6$$

从而

$$\frac{m}{3n-6}\geqslant 1$$

此式表示非平面图G分为最大平面子图的最少个数，但不一定是整数，为此取

$$\theta(G)=\left\lceil\frac{m}{3n-6}\right\rceil$$

若图不含 K_3, 由式（4.6），则平面图有

$$m\leqslant 2n-1$$

即

$$0<\frac{m}{2n-4}\leqslant 1$$

因此不含 K_3 的非平面图，有

$$\theta(G)\geqslant\left\lceil\frac{m}{2n-4}\right\rceil$$ ∎

必须指出，（4.9）和（4.10）式给出的$\theta(G)$的下界不一定准确。例如对基本非平面图 $K_{3,3}$，已知$m=9$，$n=6$，则由式（4.9）有

$$\theta(K_{3,3})\geqslant\left\lceil\frac{9}{3\times 6-6}\right\rceil=1$$

然而显然可知 $\theta(K_{3,3})=2$

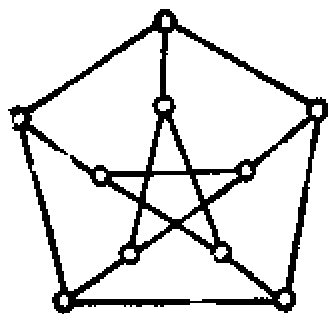


图 4.42

又如图 4.42 所示的 *Petersen* 图是一个非平面图，它的厚度 $\theta=2$，但由(4.9)式亦有

$$\left\lceil\frac{15}{3\times 10-6}\right\rceil=\left\lceil\frac{15}{24}\right\rceil=1$$

下面讨论完全图的厚度，为此先证明一个引理

引理 4.1 对任何实数b和满足 $0<\delta<1$ 的正数δ，有

$$\lceil b\rceil\geqslant\lfloor b+\delta\rfloor$$

式中$\lfloor x\rfloor$表示小于等于x的最大整数

证：若 b 为整数，则

$$\lceil b \rceil = b, \lfloor b+\delta \rfloor = b$$

因而

$$\lceil b \rceil = \lfloor b+\delta \rfloor$$

若 b 不是整数，不妨设 $n-1<b<n$ n 为整数，则

$$\lceil b \rceil = n, \lfloor b+\delta \rfloor = n-1 \text{ 或 } n,$$

因而

$$\lceil b \rceil \geqslant \lfloor b+\delta \rfloor \qquad \blacksquare$$

定理 4.9 除 $\theta(K_9)=\theta(K_{10})=3$ 外，对任何 $K_n(n>3)$，都有

$$\theta(K_n) \geqslant \left\lceil \frac{n+7}{6} \right\rceil \tag{4.11}$$

证：对于完全图 K_n，有

$$m=\frac{1}{2}n(n-1)$$

代入（4.9）式有

$$\theta(K_n) \geqslant \left\lceil \frac{n(n-1)}{6(n-2)} \right\rceil$$

$$\geqslant \left\lfloor \frac{n(n-1)}{6(n-2)}+1-\frac{1}{3n-6} \right\rfloor = \left\lfloor \frac{n+7}{6} \right\rfloor \qquad \blacksquare$$

§4.6 对 偶 图

平面图还有一个重要的特征，就是存在对偶图，非平面图就没有这个特征。下面先介绍对偶图的概念。我们约定，下面所讨论的图不限于简单图。即图可以有自环和平行边。

定义 4.9 设 $G=(V, E)$ 是可平面图的一个平面表示图，则满足下列条件的图 $G^*=(V^*, E^*)$ 称为图 G 的对偶图。

1．在 G 的每一个面 R_i 内，有一点且仅有一点 $v_i\in V^*$。

2．对 G 的任意两个面 R_i 和 R_j 的公共边 e_k，存在且仅存在一条边 $e_k^*\in E^*$ 使 $e_k^*=(v_i^*, v_j^*)$，且与边 e_k 交叉。

3．当且仅当 e_k 是 G 的唯一一个面 R_i 的边界时，v_i^* 有且仅有一个自环 $e_k^*\in E^*$ 且与 e_k^* 相交。

这一定义，实际上也给出了对偶图的构造过程，称之为 D—过程。

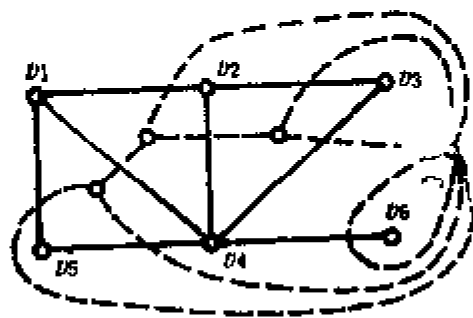


图 4.43

如图 4.43，其中平面图 G 用实线表示，它的对偶图 G^* 用虚线表示。

用 D—过程定义的对偶图称为几何对偶，这个定义也称为对偶图的几何定义。根据这一定义可得出对偶图的下列特征。

1．任何平面图的对偶图 G^* 必然是连通图。这是显然的，因为平面图只有一个无限面，所以有限面中的结点与各有限面中的结点必然是连通的。

2．G和G^*都是平面图。

3．当G是连通平面图时，G和G^*互为对偶图。

4．若n，m，r分别表示G中的结点数、边数和面数，而n^*、m^*、r^*分别表示G^*中对应的各数，则

$$m^*=m$$

$$n^*=r$$

$$r^*=n$$

5．G中回路的边在G^*中的对应边构成G^*的割集，而G的割集在G^*中的对应边构成G^*的回路，反之亦然，且对应的割集和回路所含的边数相同。

应该特别指出，在定义4.9中，我们称G^*是可平面图的一个平面表示图G的对偶图，原因在于一个给定的可平面图可以有不同的平面表示法，即可以通过球极平面投影的变换获得不同的平面表示图，这些平面表示图都是拓朴等价的，即彼此都是同构的，然面按照D—过程构造它们的对偶图，各对偶图却不是同构的。例如图4.44中，实线表示的图是图4.43可平面图的另一平面表示法，虚线表示的图是按照D—过程得到的对偶图，可以看出它与图4.43的对偶图并不同构。就是说两个平面图是同构的，它们的对偶图却不同构。

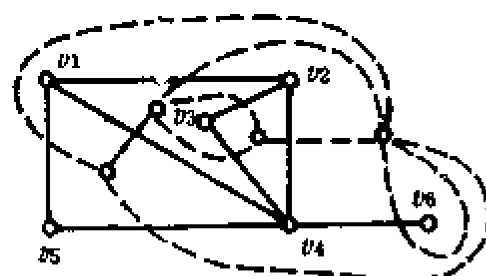
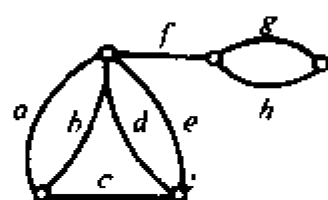


图 4.44

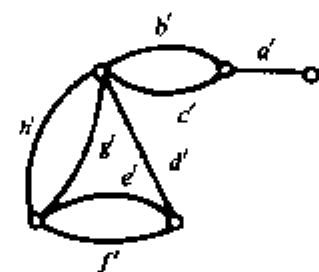
一个可平面图采用不同的平面表示，却得到不同的对偶图，虽然它们不是同构的，但进一步分析可以得出它们是2—同构的。为了说明这点，下面先介绍2—同构的概念。

定义 4.10 如果两个图G_1和G_2的边有一一对应的关系，而且对应边构成的回路也一一对应，则称G_1和G_2是互为2—同构的。

图4.45(a)和(b)的两个图G_1和G_2是2—同构的，表4.1和表4.2分别列出了它们的边和回路的对应关系。



(a) G_1



(b) G_2

图 4.45

表 4.1

G_1 中的边	G_2 中的边
a	h'
b	g'
c	d'
d	f'
e	e'
f	a'
g	b'
h	c'

表 4.2

G_1 中的回路	G_2 中的回路
ab	h'g'
de	e'f'
gh	b'c'
acd	h'd'f'
ace	h'd'e'
bcd	g'd'f'
bce	g'd'e'

如何判定两个图是否是2—同构的，下面我们不加证明地引进一个判别定理。

定理 4.10 两个图是2—同构的充分必要条件是它们具有相同的基本割集矩阵。

例 4.7 如图4.45(a)(b)的两个图G_1和G_2，取G_1的一棵生成树

$$T(G_1)=\{a,\ d,\ f,\ g\}$$

得到基本割集如下

$$K(a)=\{a,\ b,\ c\}$$
$$K(d)=\{d,\ c,\ e\}$$
$$K(f)=\{f\}$$
$$K(g)=\{g,\ h\}$$

在 G_2 中边与 $T(G_1)$ 对应的生成树为

$$T(G_2)=\{h',\ f',\ a',\ b'\}$$

得到基本割集如下

$$K(h')=\{h',\ g',\ d'\}$$
$$K(f')=\{f',\ e',\ d'\}$$
$$K(a')=\{a'\}$$
$$K(b')=\{b',\ c'\}$$

可见两个基本割集也是一一对应的，所以 G_1 和 G_2 是 2 一同构图。

这一定理具有重要的理论意义，但是用来判定两个图是否 2 一同构，还是相当麻烦的，这是因为和定义 4.10 提出的条件一样，首先要找出两个图一一对应的边，当图比较复杂时，要做到这点并非易事。下面我们介绍判定 2—同构的另一方法，称为怀特耐（Whitney）几何判别法。

定理 4.11 （Whitney）图 G_1 和 G_2 如果重复应用下列操作Ⅰ或操作Ⅱ而成为同构图，则 G_1 和 G_2 是 2 一同构的。

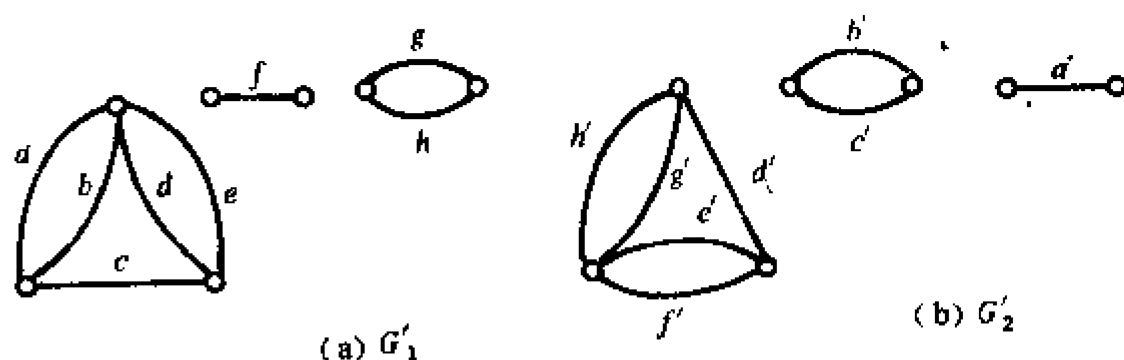
操作Ⅰ　如图存在割点，则在割点处将图切开，使图分成两个或更多的子图。

操作Ⅱ　若图或它切开后的某个子图由两部分 H 和 $\bar{H}$ 组成，且二者只有两个公共点，例如是结点 a 和 b，则把 H 中这两点的标号互换，即结点 a 换成标号 b，结点 b 换成标号 a。（相当于将图 H 绕 ab 的中垂线翻转 180 度）

证： 因为操作Ⅰ或操作Ⅱ都没有改变图的边数和回路的数目，通过两种操作 G_1 和 G_2 成为同构，说明未施行两种操作之前，它们的边有着一一对应的关系，而且对应边构成的回路也是一一对应的，根据定义，G_1 和 G_2 是 2 一同构图。 ▮

例 4.8 用怀特耐法判定图4.45(a)，(b)的图 G_1 和 G_2 是 2—同构的，步骤如下。

应用操作Ⅰ，分别将图 G_1 和 G_2 从割点处切开，得到图 G_1' 和 G_2'，如图 4.46 的(a)和(b)所示，显然 G_1' 和 G_2' 同构。



(a) G_1'　(b) G_2'

图 4.46

例 4.9 图 4.47(a)和(b)的图 G 和 G' 是彼此 2 一同构的。因为按如下步骤可以使它们成为同构图。

第一步　应用操作Ⅰ分别将图G和G'从割点处切开，得到图G_1和G_1'，如图4.48的(a)和(b)所示。

第二步　图4.48(a)中的子图G_2，由两部分H和$\bar{H}$组成，且二者只有两个公共点1和2，如图4.49(a)，应用操作Ⅱ，将H的结点标号1和2互换，得到图4.49(b)

第三步　图4.49(b)中的子图G_3也由两部分H'和$\bar{H}'$组成，如图4.50的(a)，应用操作Ⅱ，将H'翻转得到图4.50(b)，可见图4.50(b)和图4.48(b)两个图是同构的。

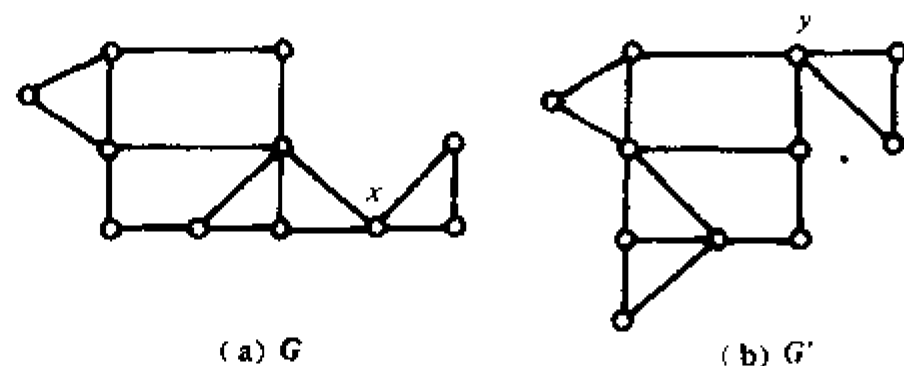


图 4.47

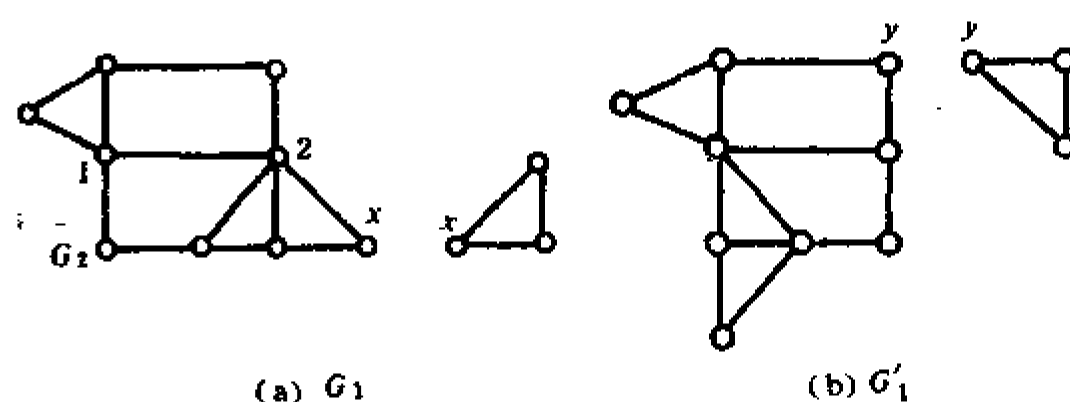


图 4.48

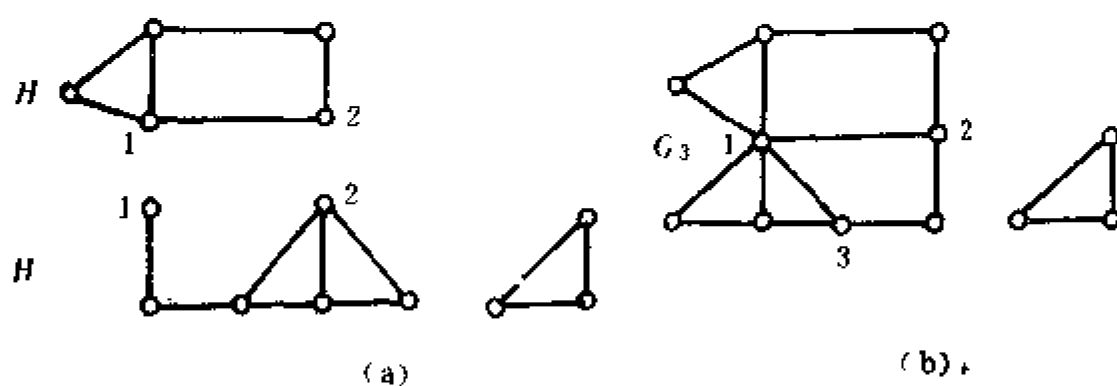


图 4.49

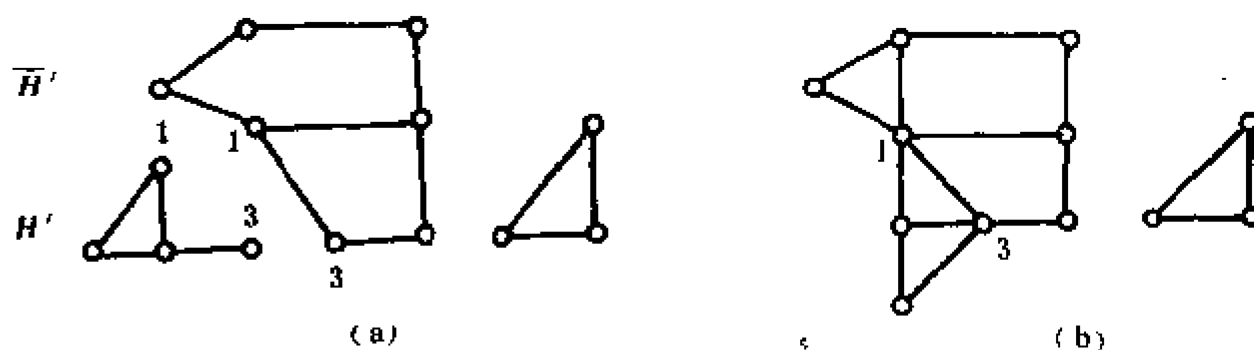


图 4.50

现在我们回过头来看图4.43和图4.44所示的两个对偶图（图中虚线所示）并将它们重画出如图4.51的(a)和(b)所示，可见如果将(a)图以1和2为公共点分为两个子图，并将其中一个翻转，则得到的图与(b)图同构，所以这两个对偶图是彼此2一同构的。



图 4.51

通过以上分析，我们得到对偶图的一些特征、下面不加证明地引入一个定理，作为这一问题的一个小结。

定理 4.12 可平面图G的所有平面表示图的对偶图，都是彼此2—同构的，而且如果G_1是G的一个对偶图，G_2与G_1 2—同构，则G_2也是G的对偶图。

根据这一定理，一个可平面图的对偶图可以不是连通图，这样的非连通对偶图用D—过程是无法获得的，由此看出对偶图的几何定义的局限性，为了弥补这一不足，人们提出了对偶的组合定义。

定义 4.11 图G_1和G_2的边一一对应，并且当且仅当C是G_1中任一简单回路的边集时与之对应的边集C^*是G_2的割集，则称G_2是G_1的对偶。

由定义可知，对偶是相互的，而在几何对偶中，必须是连通图才具有这种相互性。此外，定义并未提到图的平面性，即组合对偶不依赖于图的平面表示，因而不存在几何对偶中的特有关系，即

$$r^* = n$$

但是从几何对偶的性质可知，两个图互为几何对偶，也必然是组合对偶。

定理 4.13 一个图有对偶的充分必要条件是它是平面图。

证： 一个图如果是平面图，则从D—过程可知它一定有对偶图，所以这里只须证明非平面图一定没有对偶图即可。由对偶的定义可知，一个图只有当它的每一个子图都有对偶时，它才有对偶，而非平面图一定含有与K_5或$K_{3,3}$二次结点内同构的子图，因此只要证明K_5或$K_{3,3}$没有对偶，即证得任一非平面图均无对偶。下面采用反证法。

设非平面图K_5有对偶K_5^*，因为K_5有

（1）10条边

（2）没有长度为2的回路

（3）没有两条边的割集

（4）只有4条边和6条边的割集

由定义4.11，对偶图K_5^*应有

（1）10条边

（2）结点的次数至少是3

（3）没有两条边的回路

（4）只有4条边和6条边的回路

现在来分析K_5^*，由（4），至少有6个结点，如K_5^*有6个结点，并有10条边，则图必然有至少一个两条边的回路或3条边的回路，与K_5^*应有的条件矛盾。如K_5^*的结点数

超过 6，由于每一结点次数至少是 3，因而边的数目必大于 10，又与(1)矛盾。所以不可能存在 K_5^*

同理，设非平面图 $K_{3,3}$ 有对偶 $K_{3,3}^*$，因 $K_{3,3}$ 有：

（1） 9 条边

（2） 没有两条边的割集

（3） 只有 4 条边和 6 条边的回路

对应地 $K_{3,3}^*$ 应有

（1） 9 条边

（2） 没有两条边的回路

（3） 没有少于 4 条边的割集

现在来分析 $K_{3,3}^*$，由（3）可知每一结点的次数至少是 4，因此至少有 5 个结点，则边的数目至少是 10，与(1)矛盾，所以不可能存在 $K_{3,3}^*$ ▮

定义 4.12 如果平面图 G 与它的对偶图 G^* 同构，则称 G 为自对偶图。

图 4.52 所示的四阶完全图 K_4 是一个自对偶图。

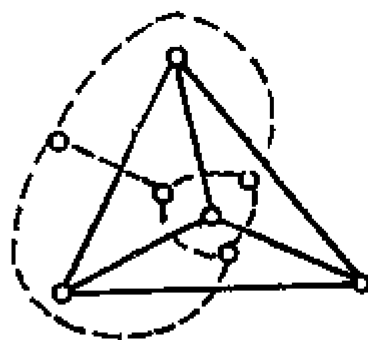


图 4.52

由定义可知，自对偶图有如下一些特点

（1） 自对偶图的结点数与面数相等，即

$$n = r$$

（2） 自对偶图的结点数与边数的关系为

$$m=2(n-1) \tag{4.12}$$

（3） 自对偶图如有自环和悬挂边，则它们的数目相等。

（4） 自对偶图如有平行边和串联边，则它们一一对应。

定理 4.14 在自对偶图 G 中，必存在次数小于 4 的结点和次数小于 4 的面。

证： 用反证法，设 G 中任一结点的次数都大于等于 4，由式 (2.1)结点的次数和等于边数的 2 倍，则有

$$\sum_{i=1}^{n} deg(v_i)=2m\geqslant 4n$$

即

$$m\geqslant 2n$$

与 (4.12) 矛盾。同理，设 G 的任一个面的次数都大于等于 4，由式 (4.1) 面的次数和等于边数的 2 倍，则有

$$\sum_{i=1}^{r} deg(R_i)=2m\geqslant 4r$$

即

$$m\geqslant 2r$$

因自对偶图有

$$n = r$$

故

$$m\geqslant 2n$$

也与式 (4.12) 矛盾 ▮

定义 4.13 不含自环和平行边的自对偶图称为简单自对偶图。

简单自对偶图除了具有一般自对偶图的特点之外，还有自身的特点。

定理 4.15 简单自对偶图的结点数 n 必大于等于 4。

证：用反证法，设 $n\leqslant 3$，由式（4.3）即得

$$m\leqslant 3n-6\leqslant 3$$

而由式（4.12）有

$$m=2(3-1)=4$$

产生矛盾，因此不存在少于 4 个结点的简单自对偶图。 ∎

定理 4.16 若 G 是有 n 个结点的连通简单自对偶图，则 G 中次数为 3 的结点至少有 4 个。

证：首先证明 G 不存在次数小于 3 的结点。用反证法，若存在结点 v_i 有

（1）$deg(v_i)=0$，则 v_i 是孤点，与 G 是连通图矛盾。

（2）$deg(v_i)=1$，则与 v_i 关联的边是悬挂边，则由自对偶图的性质（3）知 G 有自环，与 G 是简单自对偶图矛盾。

（3）$deg(v_i)=2$，则与 v_i 关联的两条边是串联边，则由自对偶图的性质（4）知 G 有平行边，也与 G 是简单自对偶图矛盾。

因此 G 中结点的次数必大于等于 3，令 K_m 表示次数为 m 的结点数目（$m=3, 4\cdots, n-1$），则有

$$K_m\geqslant 0$$

和

$$K_3+K_4+K_5+K_6+\cdots+K_{n-1}=n \qquad \text{(A)}$$

由式（2.1）结点的次数和等于边数的 2 倍，即

$$\sum_{i=1}^{n} deg(v_i)=3K_3+4K_4+5K_5+6K_6+\cdots+(n-1)K_{n-1}=2m$$

由式（4.12），上式可写成

$$3K_3+4K_4+5K_5+6K_6+\cdots+(n-1)K_{n-1}=4(n-1) \qquad \text{(B)}$$

令 4×（A）-（B）即得

$$K_3-[K_5+2K_6+3K_7+4K_8+\cdots+(n-5)K_{n-1}]=4$$

因 $K_m\geqslant 0$，即

$$[K_5+2K_6+3K_7+4K_8+\cdots+(n-5)K_{n-1}]\geqslant 0$$

因此必有

$$K_3\geqslant 4$$

即次数为 3 的结点至少有 4 个。 ∎

推论：简单自对偶图 G 不存在欧拉回路和欧拉通路。

欧拉回路和欧拉通路的定义见第八章。

习题与思考题

1．试将图 4.3 的平面图通过球极平面投影成为另一平面图，使原来的有限面 R_2 成为无限面。

2．证明图 4.53（a）和（b）可由球面上同一个图以不同方式通过球极平面投影到平面上而得。

3．图 4.54 中哪些是平面图？哪些是非平面图？并将平面图画在平面上。

4．图 4.55 的平面图各有几个面？并求出各图面的次数和。



图 4.53

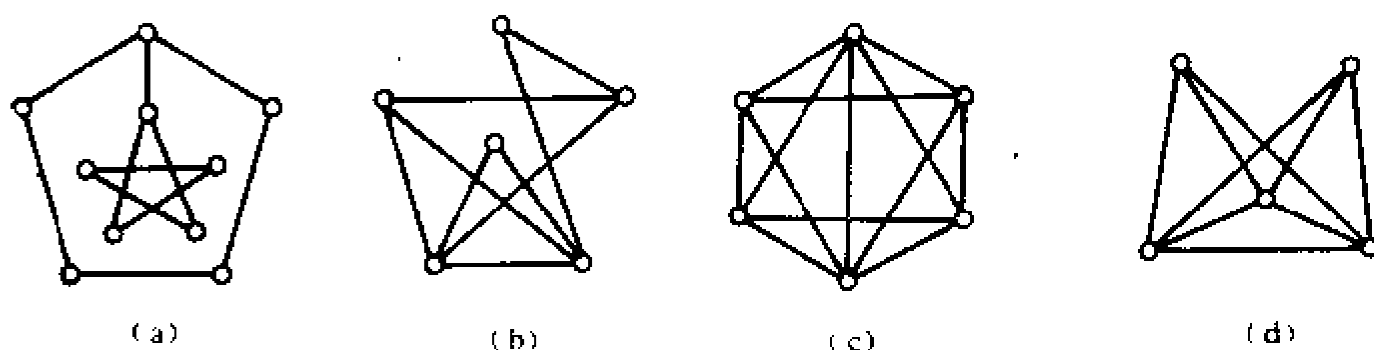


图 4.54

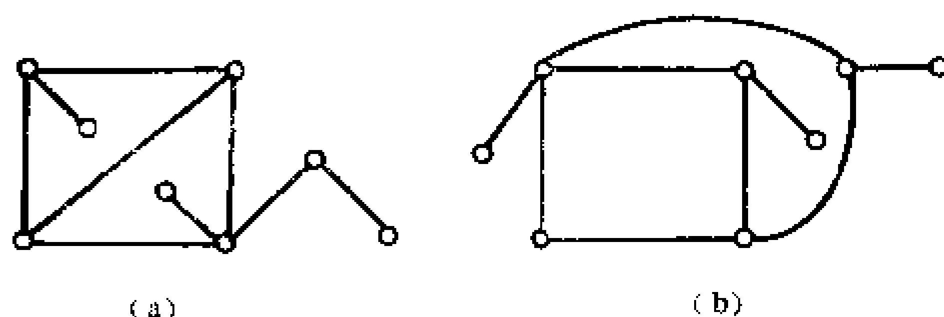


图 4.55

5．一个有 n 个结点 m 条边的简单平面图 G，若它的每一个面的次数至少是 $k(k>3)$ 证明

$$m \leqslant \frac{k(n-2)}{k-2}$$

6．证明：少于 30 条边的简单平面图必有一个结点的次数小于等于 4 。

7．证明：在有 7 个结点 15 条边的简单平面图中，每个面的次数都是 3 。

8．画出所有具有 6 个结点的非平面图，使得任何两个图都不同构。

9．设一个可平面图 G 有 28 个结点，将它嵌入平面后每个面的次数均为 4 ，试求图的边数和面数。

10．证明：任一连通平面图次数为奇数的面的数目必为偶数。

11．设 G 是一简单平面图，证明：

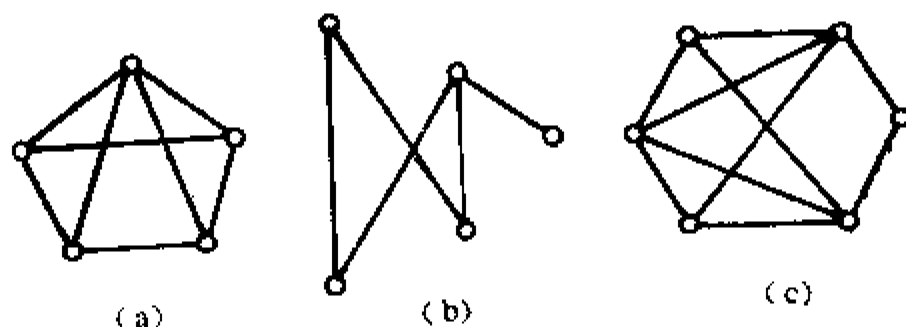
（a）若 G 中各结点的最小次数是 4 ，则至少有 6 个结点的次数不超过 5

（b）若 G 中各结点的最小次数是 5，则至少有 12 个结点的次数是 5

12．证明：若图 G 是结点数 $n \geqslant 11$ 的简单平面图，则其补图 $\bar{G}$ 是非平面图。

13．试画出一个结点数 $n=8$ 的简单平面图 G，使得它的补图 $\bar{G}$ 也是平面图。

14．图 4.56 中各图是否是极大平面图？如果不是，试在图中添加边使它成为极大平面图。

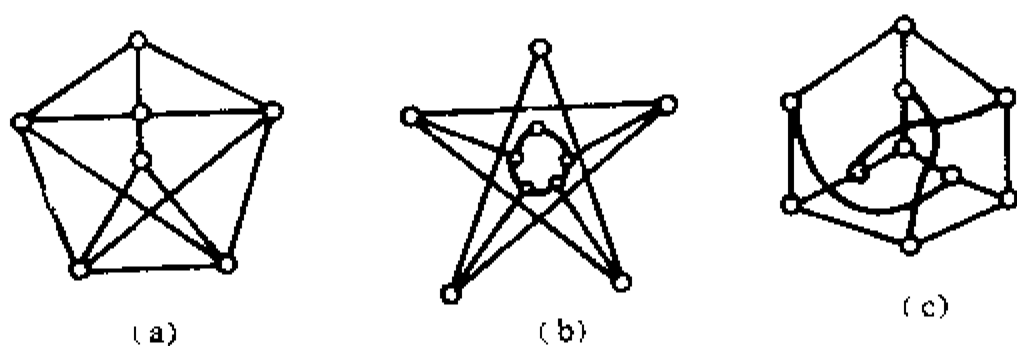


(a)　(b)　(c)

图 4.56

15．证明：任一极大平面图面的数目必为偶数。

16．试用库拉托夫斯基定理说明图 4.57 所示各图是非平面图。



(a)　(b)　(c)

图 4.57

17．如图 4.58 所示的图 G，取回路 $C=(v_1, v_2, v_3, v_4)$ 作为子图 G_1，试求出 G 相对 G_1 的片。

18．试用 §4.4 的简化步骤对图 4.59 的图 G 进行简化，并判定 G 是否是平面图。

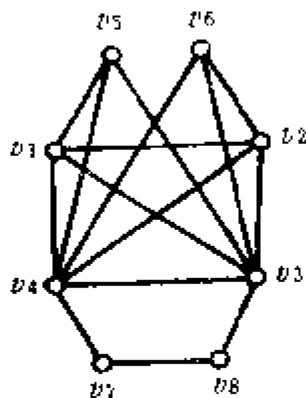


图 4.58

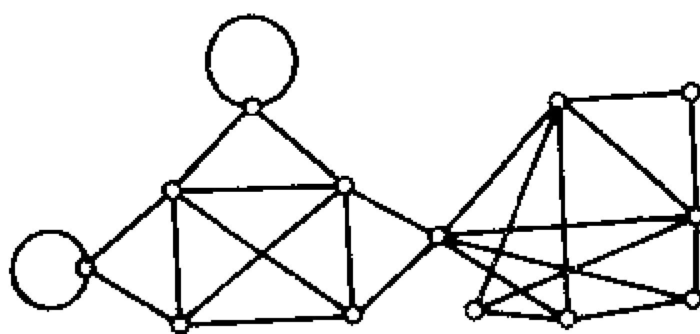
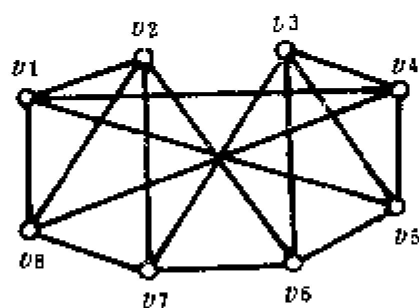
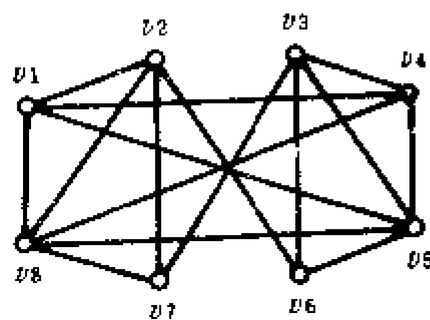


图 4.59

19．试用 $D.M.P$ 算法判断图 4.60(a)和(b)两个图的平面性。



(a) G_1



(b) G_2

图 4.60

20．试画出 3 个边不重的平面图 G_1、G_2 和 G_3，使它们都是 10 阶完全图 K_{10} 的生成

子图，且满足

$$G_1 \cup G_2 \cup G_3 = K_{10}$$

21．试用画图的方法说明 8 阶完全图 K_8 的厚度是 2 。

22．试用D—过程画出图 4.61(a)、(b)两个图的对偶图。

23．试找出图 4.62(a)和(b)两个图边的一一对应关系以及相应回路之间的一一对应关系，从而表明两个图是 2 —同构的。

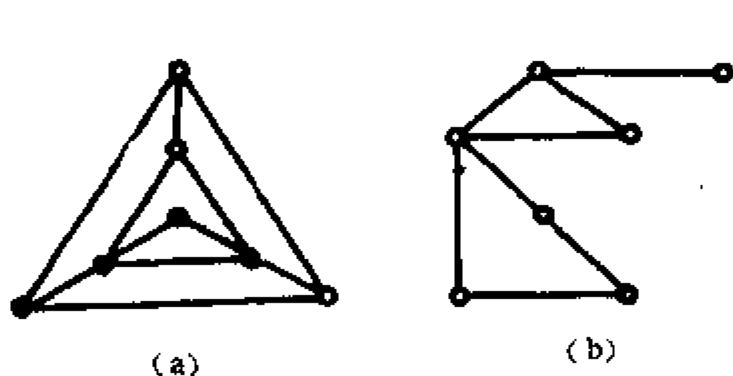


图 4.61

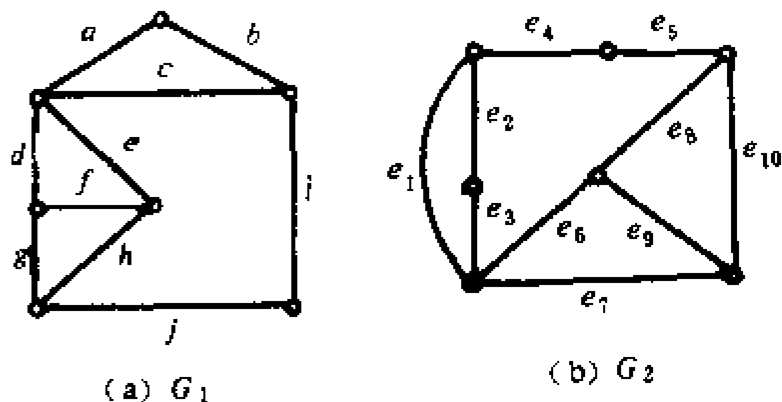


图 4.62

24．试用怀特耐几何判定法判别图 4.63 所示的两个图 G_1 和 G_2 是否是 2 —同构的。

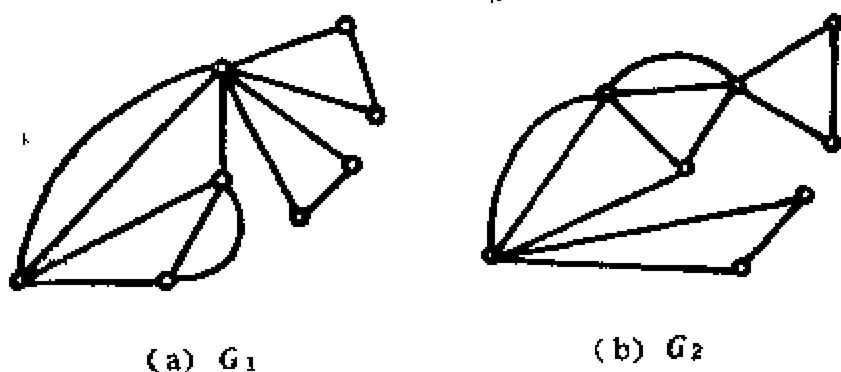


图 4.63

25．说明一个平面图G中构成一棵生成树的边对应于其对偶图G^*中构成弦的边。

26．证明：若G是不可分平面图，且所有结点的次数不小于 3 ，则G的对偶图G^*是简单图。

27．证明：若G是 3 次正则的连通平面图，则G的对偶图G^*的每个面的次数都是3，且面数为 $2(n-2)$。

28．试对 $n \geq 4$ 的每个 n 值都求出一个在 n 个结点上的自对偶的平面图。

29．证明：同构的平面图的几何对偶图必有相同的面数。

30．试举例说明命题"平面图G有次数为 1 的结点，则其对偶图G^*有自环，若G有次数为 2 的结点，则G^*有平行边"的逆命题不成立。

31．证明：不存在这样的平面图，它有 5 个面，且任二面间都至少有一条公共边。

第五章 偶图与匹配问题

§5.1 偶图的定义及性质

定义 5.1 设无向图$G=(V,E)$的结点集合V能分成两个子集V_1和V_2，满足

$$V_1\cap V_2=\phi, V_1\cup V_2=V$$

且对任意一条边$e=(v_i, v_j)\in E$，均有

$$v_i\in V_1 \quad 和 \quad v_j\in V_2$$

则称G为偶图（或称二分图或二部图）。并称V_1和V_2为G的互补结点子集。

例 5.1 图5.1是一个偶图，它的互补结点子集为

$$V_1=\{v_1, v_2, v_3, v_4\}$$

$$V_2=\{v_5, v_6, v_7\}$$

从定义可以看出偶图的特点，即图的每一条边都跨接在两个互补结点子集上，而结点子集内部任意两个结点都不邻接。由定义亦可知，偶图可以是连通图，也可以不是连通图。

定义 5.2 如果偶图G的互补结点子集V_1中的每一结点都与V_2中的所有结点邻接，则称G为完全偶图。

常用$K_{m,n}$表示一个完全偶图，其中m，n分别表示V_1和V_2中的结点数，即$|V_1|=m$, $|V_2|=n$。显然完全偶图$K_{m,n}$有$m\times n$条边。

图5.2中的（a）是完全偶图$K_{3,4}$，而（b）则表示完全偶图$K_{3,3}$。

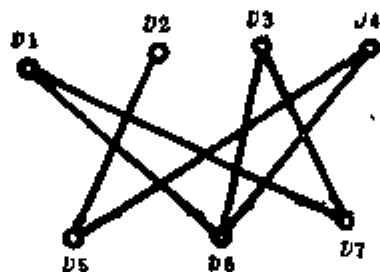


图 5.1

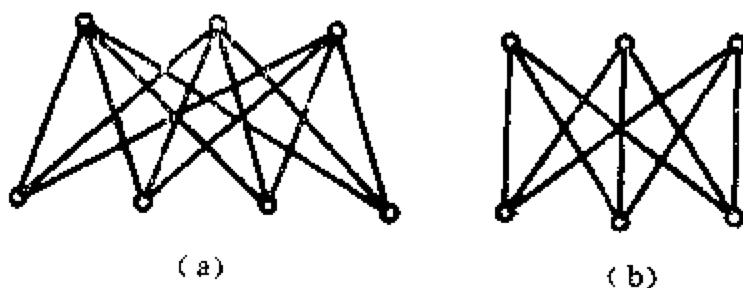


图 5.2

一个图如果具有上面的形状，很容易判定它是一个偶图，有些图虽然不是上面的形状，但是经过改画以后能成为上面的形状，仍可判定是偶图。

例如图5.3的（a）经改画后可成为（b），（c）经改画后可成为（d），因而可以判定（a）和（c）都是偶图，但图（e）无论怎样改画，都不能成为上面偶图的标准形状，因而不是偶图。

显然，根据定义或用直观的改画方法判定一个图是否是偶图是不方便的，因此有必要寻求一个行之有效的判别定理。

定理 5.1 当且仅当无向图G的每一个回路的次数均为偶数时，G才是一个偶图。

说明：如果G无回路，相当于任一回路的次数为0，0视为偶数。

证：必要性，即若G是偶图，则G的任一回路的次数为偶数。

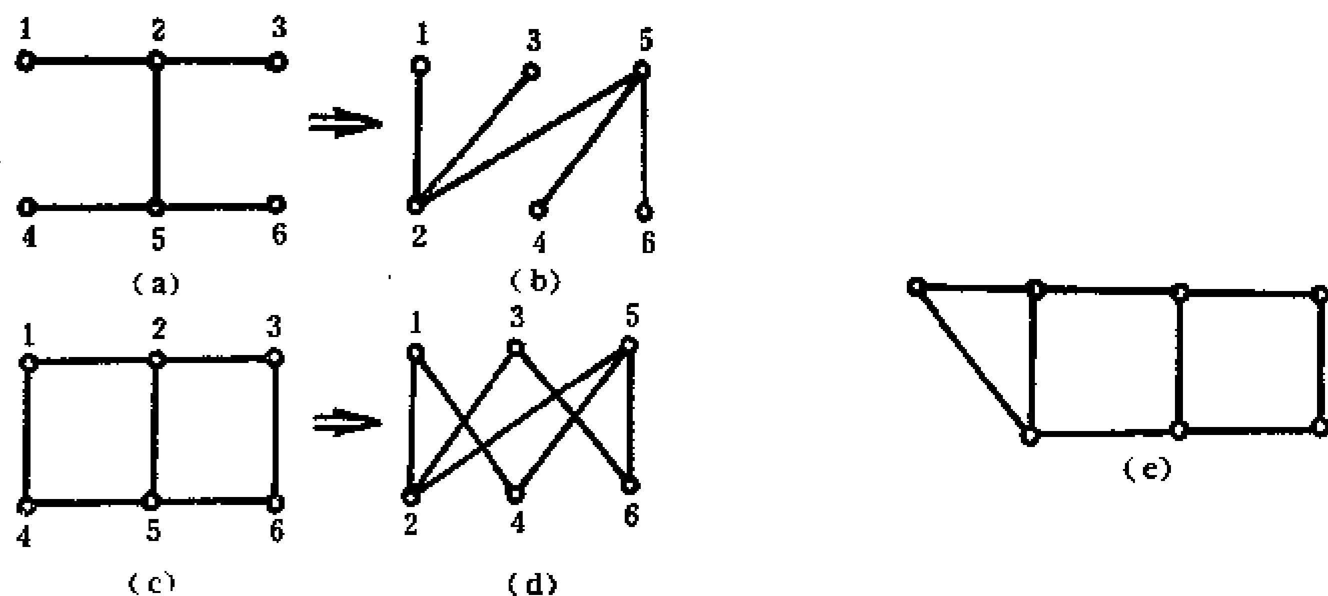


图 5.3

设G中任意一个长度为m的回路$C=v_{i_0}, v_{i_1}, v_{i_2}\cdots, v_{i_{m-1}}, v_{i_0}$。因$G$是偶图，因此可将$V$分为两个互补结点子集$V_1$和$V_2$，因$v_{i0}$和$v_{i1}$是邻接的，设

$$v_{i_0}\in V_1$$

必有

$$v_{i_1}\in V_2$$

同理可得

$$v_{i_2}, v_{i_4}, \cdots, v_{i_{m-2}}\in V_1$$

$$v_{i_3}, v_{i_5}, \cdots, v_{i_{m-1}}\in V_2$$

从结点的下标可以看出，下标为偶数的结点属于V_1，而下标为奇数的结点属于V_2，今$v_{i_{m-1}}$属于V_2，故$m-1$为奇数，即m为偶数。

充分性，即G的任一回路的次数为偶数时，G必是偶图。分两种情况进行讨论

（1） G是连通图

将图的结点集合V按下列定义分为两个子集：

$$V_1=\{v_i|v_i\text{ 与某一指定结点 }v_0\text{ 的距离为偶数}\}$$

$$V_2=V-V_1$$

下面证明V_1和V_2就是V的两个互补结点子集。

任取图G的一条边$e=(v_i, v_j)$，如果e的两个端点v_i和v_j都在V_1中，如图5.4所示的那样得到一个回路$C=v_i, \cdots, v_0, \cdots, v_j, v_i$，因$v_i, v_j\in V_1$，按定义$v_i$和$v_j$到$v_0$的距离都是偶数，再加上边$e$，故回路$C$的长度为奇数，与题设矛盾，说明$v_i$和$v_j$不可能都处于$V_1$中。

如果任意边e的两个端点v_i和v_j都在V_2中，则由定义v_i和v_j到v_0的距离都是奇数，再加上边e，则回路C的长度仍为奇数，也与题设矛盾，说明v_i和v_j也不可能都处于V_2中。因此只有唯一一种可能，即e的两个端点，一个在V_1中而另一个在V_2中，由于e是任意一条边，根据偶图的定义，G是偶图。

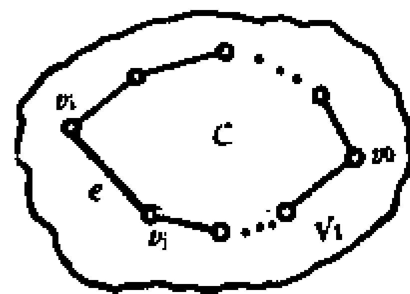


图 5.4

（2） G是非连通图

此时可分片讨论，对G的每个连通分量应用上面的证明，然后合并起来，即可得证。

应用定理检查图5.3的（a）和（c），图中回路的次数都是偶数，故都是偶图，而（e）图存在次数为3的回路，故不是偶图。

推论：任何一棵树或森林都是偶图。

例 5.2 图5.5(a)称为赫尔施尔（Herschel）图，由图可见图的任一基本回路的次

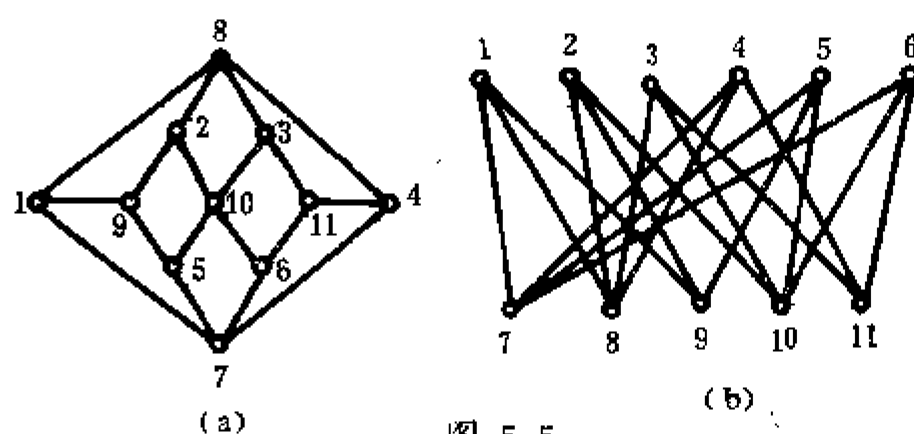


图 5.5

数都是偶数，这些回路的环和所构成的回路也必然是偶次的，所以它是偶图，(b)是将它画成偶图的标准型。

§5.2 匹配的概念

定义 5.3 在无向图$G=(V, E)$中，对边集E的任一子集$M\subseteq E$，如果M中任意两条边都不相邻，则称M为图G的一个匹配(或对集)。

所谓两条边不相邻，即两条边无公共端点。

如图5.6的图G，下列边集都是G的匹配。

$$M_1=\{e_1\}$$

$$M_2=\{e_1, e_5\}$$

$$M_3=\{e_1, e_5, e_{10}\}$$

$$\vdots$$

$$M_k=\{e_4, e_6, e_8\}$$

显然，图G的任一匹配M的任一子集$M'\subseteq M$，仍是G的匹配。

G中属于M的边称为匹配边，匹配边的两个端点互为匹配点，匹配边的所有结点称为关于M饱和点，否则称为非饱和点。匹配M的基数（即M中边的数目）记作$|M|$。

设M是G的一个匹配，若不存在另一匹配M'，满足$|M'|>|M|$，则称M是G的最大基数匹配。若G的每个结点都是M饱和点，则称M是G的完美匹配。显然，完美匹配一定是最大基数匹配。

图5.7(a)和（b）分别给出了图的最大基数匹配和完美匹配。(图中粗线表示匹配

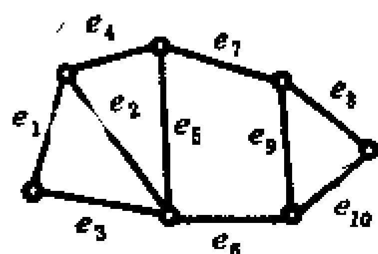


图 5.6

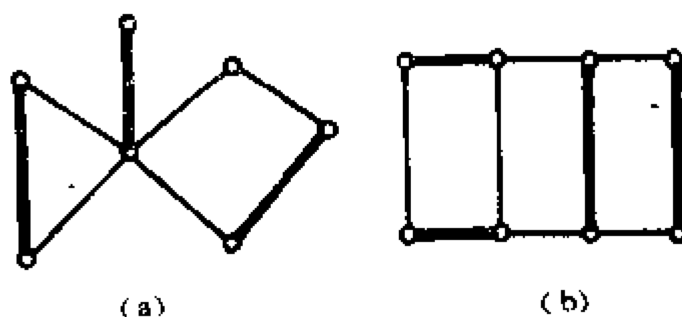


图 5.7

边)。由图可见，一个图G的最大基数匹配或完美匹配不是唯一的，而图具有偶数个结点，则是存在完美匹配的必要条件。

设M是图G的一个匹配，若G中存在一条基本路径P，路径的边是由属于M的匹配边和不属于M的非匹配边交替出现组成，则称P为交替路。若P的两个端点都是M的非饱和点，则称这条交替路为可增广路。

例如图 5.8(a)中图G的一个匹配M为（图中粗线 所示）

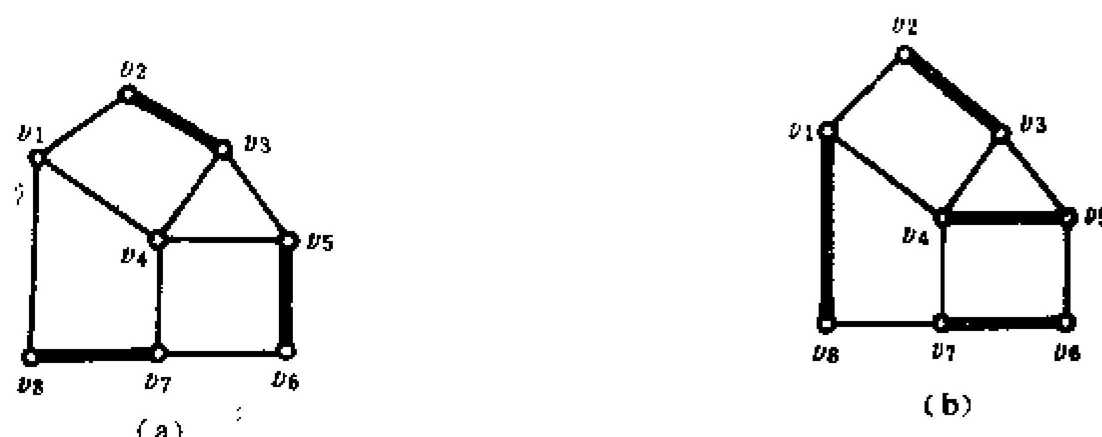


图 5.8

$$M=\{(v_2, v_3), (v_5, v_6), (v_7, v_8)\}$$

则

$$P_1=v_1v_2v_3v_5$$

是一条交替路，而

$$P_2=v_4v_5v_6v_7v_8v_1$$

则是一条可增广路，如用$E(P)$表示路径P的边集，则

$$E(P_2)=\{(v_4, v_5), (v_5, v_6), (v_6, v_7), (v_7, v_8), (v_8, v_1)\}$$

如果将可增广路P_2上的匹配边改为非匹配边，非匹配边改为匹配边，不在P_2上的匹配边保持不变，则匹配变为M'，如图 5.8(b)所示，可以看出M'的边数比M的边数增加1，即

$$|M'|=|M|+1$$

上面由M变为M'的变换，实际是对M和$E(P_2)$进行环和的结果，即

$$M'=M\oplus E(P_2)$$

定理 5.2 图G的匹配M是最大基数匹配当且仅当G不含M可增广路。

证：如果G含有M可增广路P，由前面的分析显然可构造一个新的匹配$M'=M\oplus E(P)$，且$|M'|>|M|$，因而M就不是最大基数匹配。

反之，如果M不是最大基数匹配，下面将证明G必含有M可增广路。

令M'是G的最大基数匹配，则$|M'|>|M|$。

设

$$G'=(V, M\oplus M')$$

则G'是G的生成子图，G'的边集是M和M'中不相同的边的并集，而且可以得到

（1）G'的边集中含M'的边的数目比含M的边的数目多

（2）G'的任一结点最多关联M'的一条边或最多关联M的一条边。

由（2）可知G'的任一结点的次数或者是0，不然就是1或2，因此G'的每个连通分支或者是由M'和M的边交替出现构成的回路，或者是由M'和M的边交替出现构成的交替路，由于G'中M'的边多于M的边，因此必有一条交替路P始于M'的边且终止于M'的

边，这条交替路的两个端点对 M' 是饱和点，对 M 则是非饱和点，因此 P 是 G 的一条可 M 增广路。

图的匹配理论是图论的一个重要组成部分，因为很多实际问题都可化为图的匹配问题进行分析和得到解决。下面将图的匹配问题分为两部分进行讨论，即偶图的匹配和一般图的匹配。

§5.3 偶图的完全匹配

所以要把偶图的匹配问题从一般图的匹配问题中抽出来，作为一个独立的内容进行讨论，一方面是由于偶图的匹配问题有它的特殊性，更为重要的是它有着更为广泛的应用，所谓人员工作安排问题，就是一个典型的例子。

设有 n 个工作人员 $x_1, x_2, \cdots, x_n$ 安排做 m 项工作任务 $y_1, y_2, \cdots y_m$，并不是任一工作人员都能从事任何一项工作，一般情况是某一工作人员只能从事其中某一项或某几项工作，于是就提出一个问题：是否可能使每一工作人员都能分到一项工作？如果不可能，怎样合理安排使尽可能多的工作人员有工作可做。

图 5.9 中，x_1, x_2, x_3, x_4, x_5 为 5 个人，y_1, y_2, y_3, y_4, y_5 为 5 项工作，x_1 能做 y_1，y_3，y_4 三项工作，则联接边 x_1y_1，x_1y_3，和 x_1y_4 如图所示，x_5 只能做 y_2 一项工作，则联接边 x_5y_2，其他情况类似，于是人员的工作安排问题就成为偶图的匹配问题。

定义 5.4 设 V_1 和 V_2 是偶图 $G=(V, E)$ 的两个互补结点子集，如果存在匹配 M，使 V_1 的所有结点都是 M 饱和点，则称 M 为从 V_1 到 V_2 的完全匹配。

完全匹配必然也是最大基数匹配。但是，一个偶图不一定存在完全匹配，例如当 $|V_2|<|V_1|$ 时，如图 5.10(a) 所示，就不可能获得完全匹配，即使 $|V_2|\geqslant|V_1|$，也不一定存在完全匹配，如图 5.10(b) 所示。所以 $|V_2|\geqslant|V_1|$ 只是完全匹配的必要条件而非充分条件。

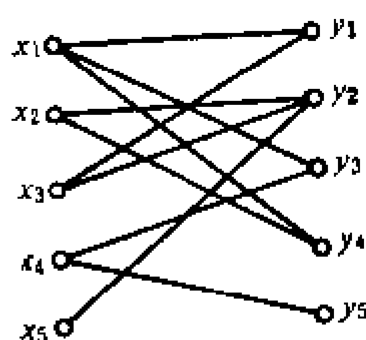


图 5.9

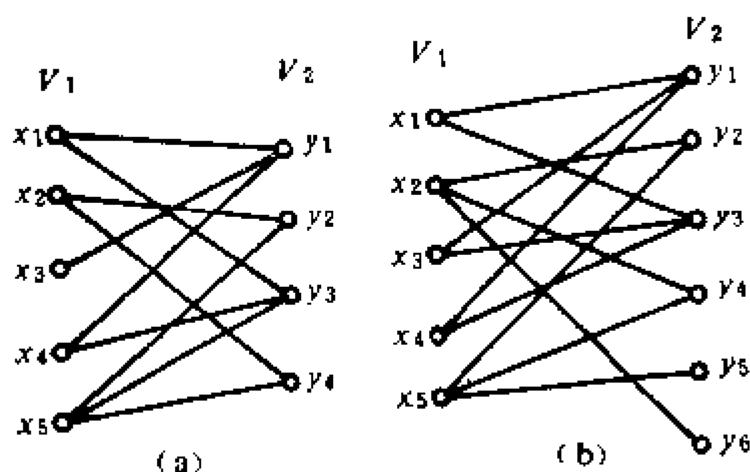


图 5.10

那么，怎样判断一个偶图是否存在完全匹配呢？下面给出的 *Hall* 定理定量地解决了这个问题。

定理 5.3 设 $G=(V, E)$ 是一个具有互补结点子集 V_1 和 V_2 的偶图，则 G 有从 V_1 到 V_2 完全匹配的充要条件是，对任意 $A\subseteq V_1$，均有

$$|N(A)|\geqslant|A| \tag{5.1}$$

式中 $N(A)\subseteq V_2$ 为与 A 中结点邻接的结点集合。

证：必要性：如果存在从 V_1 到 V_2 的完全匹配 M，则 V_1 中每个结点都是 M 饱和点，因此 V_1 中任一子集 A 中的结点在 M 下和 $N(A)$ 中的结点配对，显然有 $|N(A)|\geqslant|A|$

充分性：即如果满足式（5.1），则存在从 V_1 到 V_2 的完全匹配。

采用反证法，设 G 不存在完全匹配，则 V_1 中 $|V_1|=n_1$ 个结点不能与 V_2 中 n_1 个不同结点配对，现在假设找到 G 的一个最大基数匹配 M^*，则 V_1 中至少有一个结点不是 M^* 饱和点（否则 M^* 就是完全匹配了），设这个结点为 v_0，我们从 v_0 开始找出所有 M^* 的交替路，并把这些交替路上的结点集合记作 S，S 中属于 V_1 的结点集合记作 A，属于 V_2 的结点集合记作 B，即 $S=A\cup B$，$A=S\cap V_1$，$B=S\cap V_2$ 如图 5.11。

根据交替路的性质可知，与 A 中结点邻接的结点一定是 M^* 饱和点，这些结点一定都在从 v_0 开始的 M^* 交替路上，因此有

$$N(A)=B \qquad (1)$$

此外，A 中除 v_0 外，它的每个结点都与 B 中的结点一一配对，因此有

$$|B|=|A|-1 \qquad (2)$$

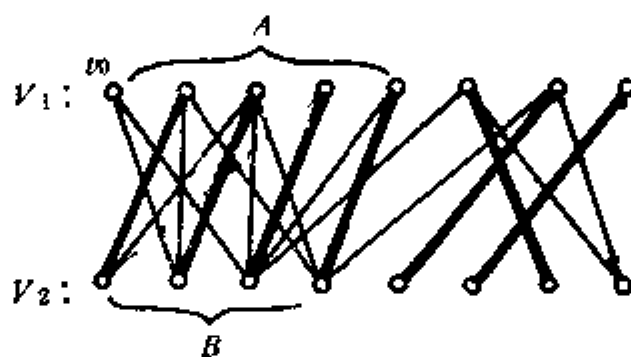


图 5.11

由（1）和（2）可得

$$|N(A)|=|A|-1<|A|$$

与题设条件（5.1）式矛盾。 ▌

式（5.1）给出的条件常称为"相异性条件"。

这一定理的证明为我们寻求偶图的完全匹配提供了一个较好的算法基础。但是应用这一定理来判别一个偶图是否存在完全匹配是很麻烦的，因为必须检查 V_1 的任一子集 A 是否满足相异性条件，如果 V_1 有 n_1 个结点，则它有 $2^{n_1}-1$ 个非空子集，显然，当 n_1 较大时，作出判断需要的工作量是很大的。下面介绍一个充分但并非必要的条件，对任一偶图，用这个定理进行判断是比较容易的，因此，在应用相异性条件之前，可应用这个定理先作判断。

定理 5.4 设 $G=(V,E)$ 是具有互补结点子集 V_1 和 V_2 的偶图，如果

（1）V_1 中每个结点至少有 t 条关联边

（2）V_2 中每个结点最多有 t 条关联边

则 G 有一个从 V_1 到 V_2 的完全匹配。（t 是一个大于零的整数）

证：因 V_1 中每个结点至少有 t 条关联边，故 V_1 中任意 k 个结点（$k=1,2,\cdots,|V_1|$）关联的边至少有 kt 条，这些边都要连接到 V_2 的结点上，而 V_2 中的结点最多与 t 条边关联，因此 V_2 中与 kt 条边关联的结点至少有 k 个，即 V_1 中任意 k 个结点至少与 V_2 中 k 个结点邻接，由式（5.1）知，存在从 V_1 到 V_2 的完全匹配。 ▌

推论：若 G 是 k 次正则偶图，则 G 有完美匹配。（$k>0$）

定理 5.4 提出的条件常称为"t 条件"。

如图 5.12 的偶图中，互补结点子集 V_1 中的每个结点至少有 3 条关联边，而 V_2 中的每个结点最多有 3 条关联边，由 t 条件可知，存在从 V_1 到 V_2 的完全匹配。

应用上面定理，可以解决所谓"学会理事长选举"问题。即设有 m 个学会，例如电子

学会、计算机学会、航空学会等，分别用 $C_1, C_2, \cdots, C_m$ 表示，这些学会的集合，称为科学技术协会，用 C 表示，即

$$C=\{c_1, c_2, \cdots, c_m\}$$

用 P_i 表示 c_i $(1\leqslant i\leqslant m)$ 的会员集合，即

$$P_i=\{a \mid a \text{ 是 } c_i \text{ 的会员}\}$$

$$P=\bigcup_{i=1}^{m} = p_1\cup p_2\cup\cdots\cup p_m$$

$$=\{a_1, a_2, \cdots, a_n\}$$

则 P 表示科学技术协会全体会员集合。如用结点 $c_1, c_2, \cdots, c_m$ 表示各学会，用结点 $a_1, a_2, \cdots, a_n$ 表示每个会员、若会员 a_i 参加学会 c_j 和 c_k，则在 a_i 与 c_j, a_i 与 c_k 之间连一条边，于是构成一个偶图如图 5.13 所示。

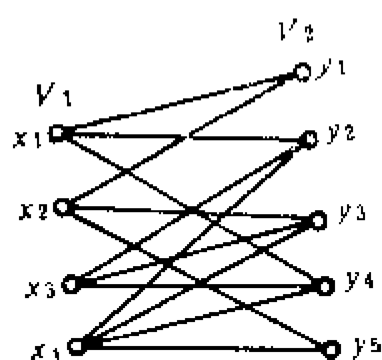


图 5.12

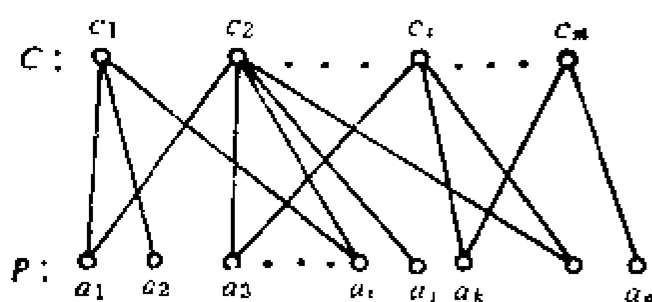


图 5.13

现在提出一个要求：要在每个学会中选出一名理事长，学会 c_i 的理事长只能从该学会的会员中选出，而且一人不得兼任两个学会以上的理事长，这个问题就归纳为在偶图中是否存在从 C 到 P 的完全匹配问题。从 t 条件可知，如果每个学会至少有 t 名会员，而每个会员最多参加 t 个学会，那么理事长的选举问题是可以解决的。

下面介绍求偶图完全匹配的一个算法。

这一算法是由 Edmonds（1965）提出的。它的基本思路是：任意找出偶图 G 的一个匹配 M，若 V_1 中的所有结点都是 M 饱和点，则 M 即是所求的完全匹配。否则在 V_1 中任找一个 M 非饱和点，比如是 v_0，以 v_0 为起点求图 G 中 M 的交替路，如所有这些交替路的终点都是 M 饱和点，说明以 v_0 为起点的交替路都不是可增广路，则图不存在完全匹配。如果有一条交替路的终点是 M 非饱和点，这条交替路就是可增广路，因而可得到边的数目多 1 条的新匹配 M'，然后再找 V_1 中一个 M' 非饱和点，重复上述过程直到 V_1 的所有结点都成为饱和点，得到的匹配即为完全匹配，否则图不存在完全匹配。

算法步骤如下：

1．任给出图 G 的一个初始匹配 M。

2．如 M 已饱和 V_1 的所有结点，则 M 即是完全匹配，计算结束，否则进行下一步。

3．找 V_1 中任一 M 非饱和点 v_0，令

$$A\leftarrow\{v_0\},\ B\leftarrow\phi$$

4．如 $N(A)=B$，则图不存在完全匹配，计算结束，否则进行下一步

5．找一结点 $y\in N(A)-B$

6．如 y 是 M 饱和点，则找出 y 的配对点 z，令

$$A \leftarrow A \cup \{z\},\ B \leftarrow B \cup \{y\}$$

转第4步，否则进行下一步

7．存在一条从 v_0 到 y 的可增广路 P，令

$$M \leftarrow M \oplus E(P)$$

转第2步。

在算法的第7步中，存在一条从 v_0 到 y 的可增广路 P，我们可以采用回溯法找出这条路径，即在集合 A 中找出与 y 邻接的结点 z，由 z 经过 M 匹配边到它的配对点 y'，如此继续下去一直回溯到起始点 v_0，所经的路径即是 v_0 到 y 的一条可增广路 P。

例 5.3 求图5.14的一个完全匹配。

1．任给出一初始匹配

$$M=\{x_1y_1,\ x_3y_5,\ x_5y_3\}$$

在图5.14中，属于 M 的边用粗线表示。

2．显然，M 未饱和 V_1 的所有结点，找出 V_1 中一未饱和点 x_2。

$$A=\{x_2\}$$
$$B=\phi$$
$$N(A)=\{y_2,\ y_3\}$$

3．因 $N(A)\neq B$，找一点

$$y_2\in N(A)-B$$

4．因 y_2 是 M 非饱和点，故存在一条从 x_2 到 y_2 的可增广路 $P=x_2y_2$，$E(P)=\{x_2y_2\}$，

$$M=\{x_1y_1,\ x_3y_5,\ x_5y_3\}\oplus\{x_2y_2\}$$
$$=\{x_1y_1,\ x_3y_5,\ x_5y_3,\ x_2y_2\}$$

于是得到一新的匹配如图5.15中的粗线所示。由于匹配 M 尚未饱和 V_1 的所有点，程序继续进行。

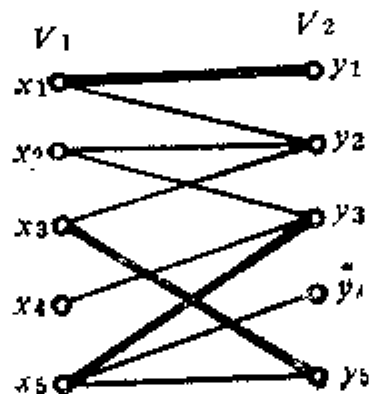


图 5.14

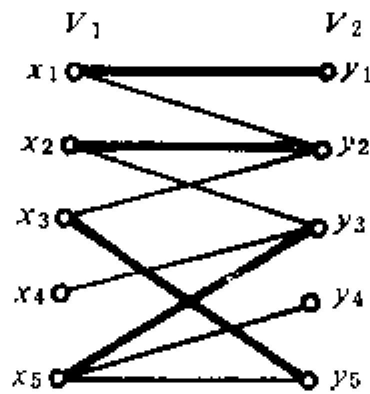


图 5.15

5．找出 V_1 中未饱和点 x_4

$$A=\{x_4\}$$
$$B=\phi$$
$$N(A)=\{y_3\}$$

6．因 $N(A)\neq B$，找一点

$$y_3\in N(A)-B$$

7．y_3 是饱和点，它的配对点是 x_5，于是

$$A=\{x_4\}\cup\{x_5\}=\{x_4,\ x_5\}$$

$$B=\{y_3\}$$
$$N(A)=\{y_3, y_4, y_5\}$$

8．因 $N(A)\neq B$, 找一点

$$y_5\in N(A)-B$$

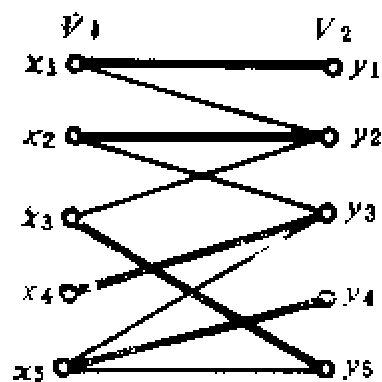


图 5.16

9．y_5 是饱和点，它的配对点是 x_3，于是

$$A=\{x_4, x_5, x_3\}$$
$$B=\{y_3, y_5\}$$
$$N(A)=\{y_2, y_3, y_4, y_5\}$$

10．因 $N(A)\neq B$, 找一点

$$y_4\in N(A)-B,$$

11．y_4 是非饱和点，因而存在一条从 x_4 到 y_4 的可增广路 P，用回溯法求出这条路径：在 $A=\{x_4, x_5, x_3\}$ 中与 y_4 邻接的点为 x_5，由 x_5 回溯到它的配对点 y_3，在 A 中找到与y_3的邻接点 x_4，回溯到了起始点 x_1，于是可增广路

$$P=x_4y_3x_5y_4$$
$$E(P)=\{x_4y_3, y_3x_5, x_5y_4\}$$
$$M=\{x_1y_1, x_3y_5, x_5y_3, x_2y_2\}\oplus\{x_4y_3, x_5y_3, x_5y_4\}$$
$$=\{x_1y_1, x_3y_5, x_2y_2, x_4y_3, x_5y_4\}$$

得到如图 5.16 所示的新的匹配（图中粗线）。这时M已饱和 V_1 的全部结点，这一匹配即为从 V_1 到 V_2 的完全匹配，计算结束。

例 5.4 试求图 5.17 所示的偶图从 V_1 到 V_2 的完全匹配

1．任给出一初始匹配

$$M=\{x_2y_2, x_3y_3, x_5y_5\}$$

如图 5.17 中的粗线所示

2．V_1 的点未完全饱和，找一非饱和点 x_1

$$A=\{x_1\}$$
$$B=\phi$$
$$N(A)=\{y_2, y_3\}$$

3．因 $N(A)\neq B$, 找一点

$$y_2\in N(A)-B$$

y_2 是饱和点，其配对点为 x_2

$$A=\{x_1, x_2\}$$
$$B=\{y_2\}$$
$$N(A)=\{y_1, y_2, y_3, y_4, y_5\}$$

4．因 $N(A)\neq B$, 找一点 $y_1\in N(A)-B$

5．y_1 是非饱和点，于是存在从 x_1 到 y_1 的可增广路 P

$$P=x_1y_2x_2y_1$$
$$E(P)=\{x_1y_2, y_2x_2, x_2y_1\}$$

6．$M=\{x_2y_2, x_3y_3, x_5y_5\}\oplus\{x_1y_2, x_2y_2, x_2y_1\}$

$=\{x_1y_2, x_2y_1, x_3y_3, x_5y_5\}$

新的匹配M如图 5.18 中粗线所示。此时 V_1 的点尚未完全饱和，程序继续进行。

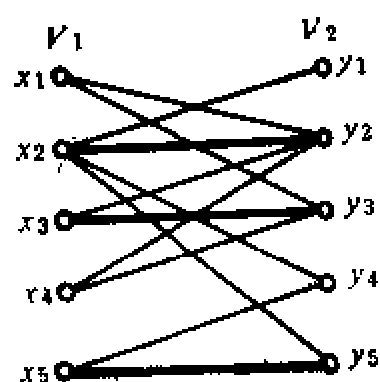


图 5.17

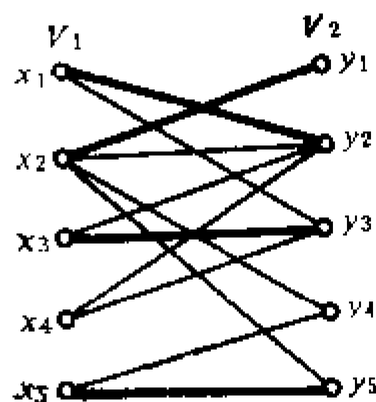


图 5.18

7．找到非饱和点 x_4

$$A=\{x_4\}$$
$$B=\phi$$
$$N(A)=\{y_2, y_3\}$$

8．因 $N(A)\neq B$，找一点

$$y_2\in N(A)-B$$

y_3 是饱和点，其配对点为 x_3

$$A=\{x_4, x_3\}$$
$$B=\{y_3\}$$
$$N(A)=\{y_2, y_3\}$$

9．因 $N(A)\neq B$，找一点

$$y_2\in N(A)-B,$$

y_2 是饱和点，其配对点为 x_1

$$A=\{x_4, x_3, x_1\}$$
$$B=\{y_3, y_2\}$$
$$N(A)=\{y_2, y_3\}$$

10．因 $N(A)=B$，故图不存在完全匹配，算法结束。

以上求完全匹配的算法只要稍加改动，即可用来求偶图的最大基数匹配，方法是将算法中的第 4 步改为

4′ 如 $N(A)=B$，则图不存在完全匹配，将 v_0 作为饱和点（或称为伪饱和点）转第 2 步，否则进行下一步。

现举一例说明如下：

例 5.5 试求图 5.19 所示的偶图的最大基数匹配。

1．给出初始匹配

$$M=\{x_1y_2, x_2y_5, x_4y_3\}$$

如图 5.19 中粗线所示

2．找到 V_1 的未饱和点 x_3，

$$A=\{x_3\}$$
$$B=\phi$$

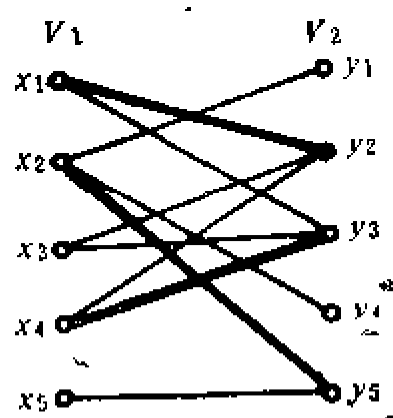


图 5.19

$$N(A)=\{y_2, y_3\}$$

3. $N(A)\neq B$, 找到点

$$y_2\in N(A)-B$$

y_2 为饱和点，其配对点为 x_1,

$$A=\{x_3, x_1\}$$
$$B=\{y_2\}$$
$$N(A)=\{y_2, y_3\}$$

4. $N(A)\neq B$, 找到点

$$y_3\in N(A)-B$$

y_3 为饱和点，其配对点为 x_4

$$A=\{x_3, x_1, x_4\}$$
$$B=\{y_2, y_3\}$$
$$N(A)=\{y_2, y_3\}$$

5. 因 $N(A)=B$, 所以偶图不存在完全匹配，将结点 x_3 当作饱和点，转算法第 2 步

6. V_1 尚未全部饱和，找到非饱和点 x_5,

$$A=\{x_5\}$$
$$B=\phi$$
$$N(A)=\{y_5\}$$

7. $N(A)\neq B$, 找到点

$$y_5\in N(A)-B$$

y_5 为饱和点，及配对点为 x_2,

$$A=\{x_5, x_2\}$$
$$B=\{y_5\}$$
$$N(A)=\{y_1, y_4, y_5\}$$

8. $N(A)\neq B$, 找到点

$$y_1\in N(A)-B$$

y_1 为非饱和点，于是得到从 x_5 到 y_1 的可增广路

$$P=x_5y_5x_2y_1$$
$$E(P)=\{x_5y_5, y_5x_2, x_2y_1\}$$
$$M=\{x_1y_2, x_2y_5, x_4y_3\}\oplus\{x_5y_5, x_2y_5, x_2y_1\}$$
$$=\{x_1y_2, x_2y_1, x_4y_3, x_5y_5\}$$

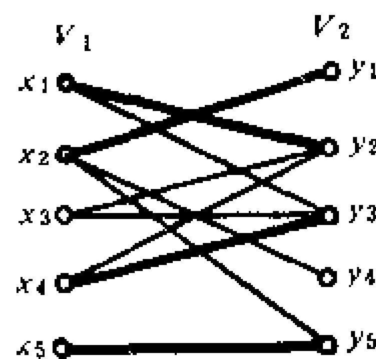


图 5.20

M的边如图 5.20 中粗线所示，由于此时 V_1 的结点已全部饱和，M即为最大基数匹配。

§5.4 偶图的最大权匹配

前面我们讨论工作人员的工作安排问题，只着眼于每一工作人员都能安排一份工作，即寻求从 V_1 到 V_2 的完全匹配，并没有考虑如何更好地发挥工作人员的专长问题，事实上，虽然某个工作人员能从事几项工作，但并非对这些工作都一样擅长，如果安排他从事

他最擅长的工作，无疑会取得更大的工作效果。如果在描述工作人员工作安排的偶图中，在边（x_i，y_j）上标上一个非负实数 $w_{ij}=w(x_i, y_j)$ 作为边的权，表示工作人员 x_i 从事 y_j 工作时取得效率的值，则寻求最大效率的工作安排就归结为寻求偶图的最大权匹配问题。如果把权作为成本，则相当于寻求偶图的最小权匹配问题。显然两者的分析方法是相同的，我们只讨论前者。

定义 5.5 设M是偶图G的一个匹配，$W(M)$表示它的边权之和，如对G的所有匹配M'，都有

$$w(M) \geqslant w(M')$$

则称M为偶图G的最大权匹配。

偶图的最大权匹配，不一定是最大基数匹配或完全匹配。例如图 5.21(a)所示的带权偶图，(b)是它的一个最大基数匹配而(c)则是它的最大权匹配。

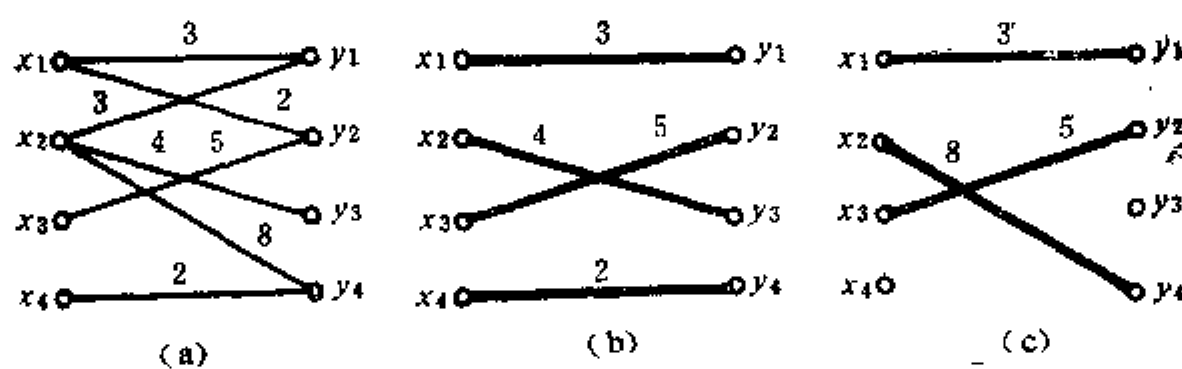


图 5.21

通常人们感兴趣的是，在完全匹配的基础上寻求最大权匹配。

定义 5.6 设M是带权完全偶图的一个完全匹配，如对偶图的所有完全匹配M'，都有

$$w(M) \geqslant w(M')$$

则称M为偶图的最优匹配。

一个完全偶图G，如果它的互补结点子集V_1和V_2的结点数均为n，即$|V_1|=|V_2|=n$，则G有$n!$个完全匹配，若求出所有这些完全匹配，然后在它们中间找出最优的，采用这种枚举法求最优匹配无疑是可以的，但当n较大时工作量将很大，下面介绍求最优匹配的一个较好算法。这一算法是采取所谓结点标记的方法，寻找图的最优匹配的。

定义 5.7 设$G=(V, E)$为一带权完全偶图，若在结点集合$V=V_1 \cup V_2$上给出一个实数函数l，使对所有的$x_i \in V_1$及$y_j \in V_2$，均有

$$l(x_i)+l(y_j) \geqslant w_{ij}=w(x_i, y_j) \tag{5.2}$$

则称l为G的一个可行结点标号，并称$l(v)(v \in V)$为结点v的标号。

例 5.6 如图 5.22(a)表示带权完全偶图$K_{3,3}$，如果给出一个标号l，使

$$l(x_1)=4,\ l(x_2)=3,\ l(x_3)=2$$
$$l(y_1)=1,\ l(y_2)=1,\ l(y_3)=2$$

则l是图的一个可行结点标号。

如果给出另一个标号l'，使

$$l'(x_1)=6,\ l'(x_2)=5,\ l'(x_3)=4$$
$$l'(y_1)=l(y_2)=l'(y_3)=0$$

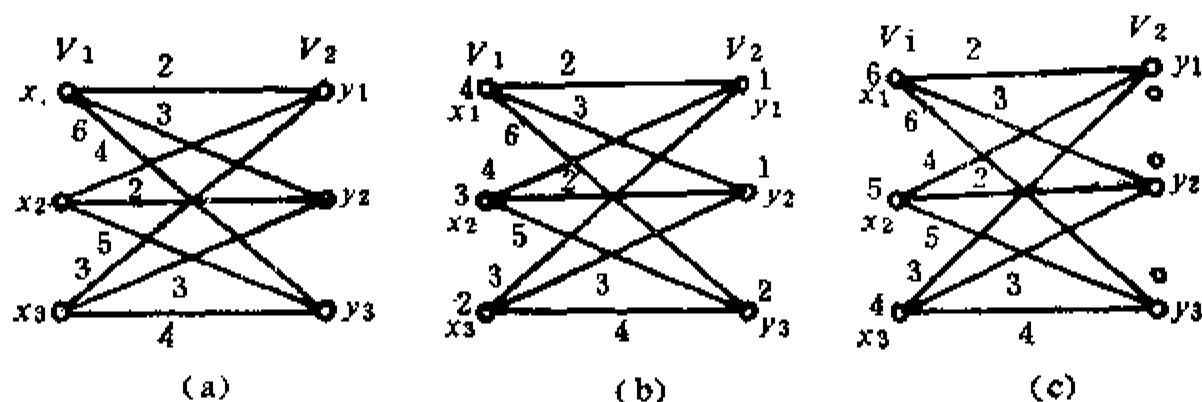


图 5.22

则 l' 也是图的一个可行结点标号。

将 l 和 l' 对结点的标号标在图的各结点上，如图 5.22 中的（b）和（c）所示，可见用来检查是否满足式（5.2），从而判定给定的 l 是否是可行结点标号，将是比较方便的。

定义 5.8 设 l 是带权完全偶图 $G=(V, E)$ 的可行结点标号，令边集

$$E_l=\{(x_i, y_j)\,|\,l(x_i)+l(y_j)=w_{ij}\}\subseteq E \tag{5.3}$$

则称生成子图 $G_l=(V, E_l)$ 为 G 对应标号 l 的等价子图。

例如在例 5.6 中，对应标号 l 和 l' 的等价子图分别如图 5.23（a）和（b）所示

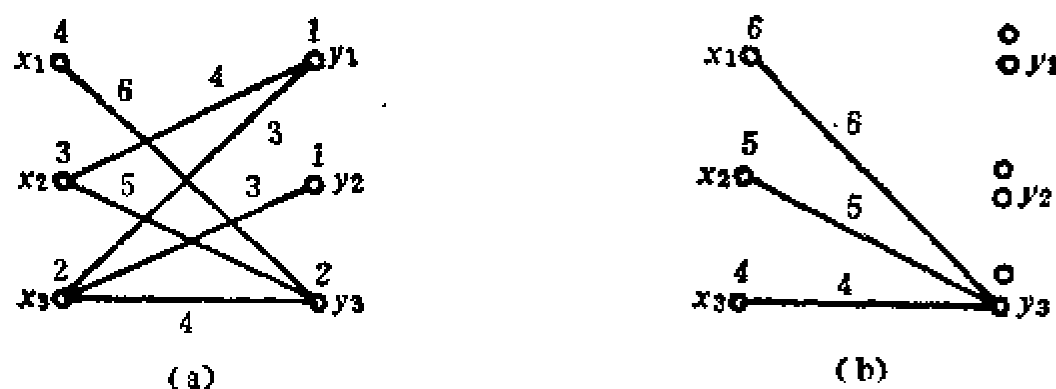


图 5.23

定理 5.5 若 l 是 G 的一个可行结点标号，如果等价子图 G_l 有一完全匹配 M^*，则 M^* 就是 G 的最优匹配。

证： 设 G_l 有一完全匹配 M^*，由于 G_l 是 G 的生成子图，故 M^* 也是 G 的一个完全匹配，即 M^* 的边饱和了 V 的全部结点，因此

$$w(M^*)=\sum_{(x_i,\,y_j)\in M^*} w(x_i, y_j)=\sum_{x_i\in V_1} l(x_i)+\sum_{y_j\in V_2} l(y_j)$$

$$=\sum_{v\in V} l(v) \tag{A}$$

此外，若 M 是 G 的任一完全匹配，则有

$$w(M)=\sum_{(x_i,y_j)\in M} w(x_i,y_j)\leqslant \sum_{x_i\in V_1} l(x_i)+\sum_{y_j\in V_2} l(y_j)=\sum_{v\in V} l(v)$$

即

$$w(M)\leqslant \sum_{v\in V} l(v) \tag{B}$$

由式（A）和（B）即得

$$w(M^*)\geqslant w(M)$$

故 M^* 是 G 的最优匹配。 ▮

这一定理将寻求 G 的最优匹配转化为寻求 G_l 的完全匹配，而寻找图完全匹配的算法，前面已经介绍过了。库恩—曼克莱斯（Kuhn—Munkres）在此基础上，提出了求最优匹配的一个较好算法。

算法的思路是：首先对带权完全偶图 G 给出一个可行结点标号 l，然后根据 l 构造等价子图 G_l，再用求完全匹配的算法求 G_l 的完全匹配（这时毋需考虑边所带的权），若 G_l 存在完全匹配 M^*，则 M^* 即是 G 的最优匹配，算法即告结束，否则修改结点标号 l，在新的结点标号下构造新的等价子图，如此不断修改，直到等价子图含有完全匹配为止。

由此可知，这一算法需要解决两个新的问题，一是最初的可行结点标号 l 怎样给出，其次，在以后的计算过程中怎样修改结点标号。

由定义可知，只要满足式（5.2）的标号 l 都是可行结点标号，l 是一个实数函数，它可以取无穷多的值，为了简单方便，通常都取如下的可行结点标号作为初始标号

$$\left.\begin{aligned} l(x) &= \max_{y\in V_2} w(x,y), & x\in V_1 \\ l(y) &= 0, & y\in V_2 \end{aligned}\right\} \tag{5.4}$$

例 5.6 给出的标号 l' 即是满足式（5.4）的可行结点标号。

至于怎样修改标号，下面的算法中将得到说明。

库恩—曼克莱斯算法

1．给出初始标号

$$l(x_i)=\max_j w_{ij},\ l(y_j)=0,\ i,j=1,2\cdots,n,$$

2．求出边集

$$E_l=\{(x_i,\ y_j)\mid l(x_i)-l(y_j)=w_{ij}\}$$

和 G_l 及 G_l 中的一个匹配 M。

3．如 M 已饱和 V_1 的所有结点，则 M 即是 G 的最优匹配，计算结束，否则进行下一步

4．在 V_1 中找一 M 非饱和点 v_0，令

$$A\longleftarrow\{v_0\},\ B\longleftarrow\phi$$

5．若 $N_{G_l}(A)=B$，则转第 9 步，否则进行下一步

6．找一结点 $y\in N_{G_l}(A)-B$

7．若 y 是 M 饱和点，则找出 y 的配对点 z，令

$$A\longleftarrow A\cup\{z\},\ B\longleftarrow B\cup\{y\}$$

转第 5 步，否则进行下一步

8．存在一条从 v_0 到 y 的可增广路 P，令

$$M\longleftarrow M\oplus E(P)$$

转第 3 步。

9．①按下式计算 a 值

$$a=\min_{\substack{x_i\in A\\ y_j\notin N_{G_l}(A)}}\{l(x_i)+l(y_j)-w_{ij}\}$$

② 修改标号：

$$l'(v)=\begin{cases} l(v)-a, & 若 v\in A \\ l(v)+a, & 若 v\in B \\ l(v), & 其它 \end{cases}$$

③ 根据 l' 求 $E_{l'}$ 及 $G_{l'}$

10 $l\longleftarrow l'$，$G_l\longleftarrow G_{l'}$，转第 6 步。

在执行这一算法时，用矩阵 $W(G)=(w_{ij})_{n\times n}$ 表示带权完全偶图是比较方便的。举例如下

例 5.7 设 x_i 为工人，y_i 为产品，$i=1,2,3,4,5$，w_{ij} 表示 x_i 制作 y_j 产品所取得的利润，得到利润矩阵如下，求一最优匹配使所得总利润最大。

$$W=\begin{matrix} & y_1 & y_2 & y_3 & y_4 & y_5 \\ x_1 & 5 & 7 & 7 & 6 & 3 \\ x_2 & 4 & 4 & 0 & 4 & 4 \\ x_3 & 4 & 6 & 6 & 3 & 0 \\ x_4 & 0 & 3 & 3 & 0 & 0 \\ x_5 & 3 & 4 & 3 & 5 & 5 \end{matrix}$$

解 1 给出初始标号 l

$$l(x_1)=\max_j w_{1j}=\max(5,7,7,6,3)=7$$

$$l(x_2)=\max(4,4,0,4,4)=4$$

$$l(x_3)=\max(4,6,6,3,0)=6$$

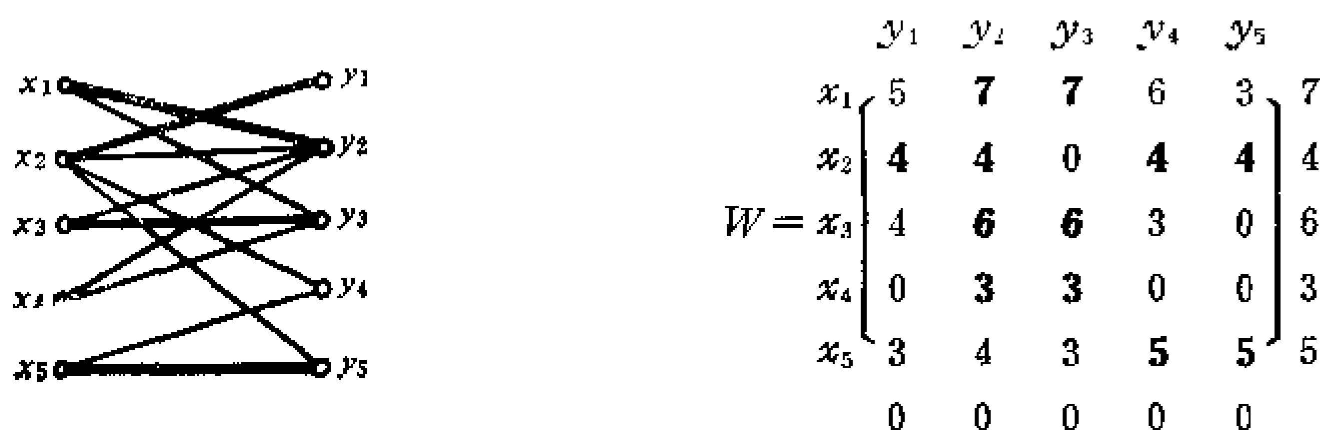
$$l(x_4)=\max(0,3,3,0,0)=3$$

$$l(x_5)=\max(3,4,3,5,5)=5$$

$$l(y_1)=l(y_2)=l(y_3)=l(y_4)=l(y_5)=0$$

2．求 E_l、G_l 及匹配 M。

将 $l(x_i)$ 的标号置于矩阵 i 行的右边，$l(y_i)$ 的标号置于矩阵 i 列的下边，并且把对应 G_l 的矩阵元素用黑体数字标出，得到下面矩阵，对应的等价子图 G_l 如图 5.24 所示。



$$W=\begin{matrix} & y_1 & y_2 & y_3 & y_4 & y_5 & \\ x_1 & 5 & \mathbf{7} & \mathbf{7} & 6 & 3 & 7 \\ x_2 & \mathbf{4} & \mathbf{4} & 0 & \mathbf{4} & \mathbf{4} & 4 \\ x_3 & 4 & \mathbf{6} & \mathbf{6} & 3 & 0 & 6 \\ x_4 & 0 & \mathbf{3} & \mathbf{3} & 0 & 0 & 3 \\ x_5 & 3 & 4 & 3 & \mathbf{5} & \mathbf{5} & 5 \\ & 0 & 0 & 0 & 0 & 0 & \end{matrix}$$

图 5.24

设在 G_l 中已求出匹配 M（图 5.24 粗线所示）

$$M=\{x_1y_2,\ x_2y_1,\ x_3y_3,\ x_5y_5\}$$

3．x_4 是 M 非饱和点，通过如下过程寻找 M 可增广路

A：$x_4\longrightarrow x_4,x_1\longrightarrow x_4,x_1,x_3$

$N(A)$：$y_2,y_3\longrightarrow y_2,y_3\longrightarrow y_2,y_3$

B：$\phi\longrightarrow y_2\longrightarrow y_2,y_3$

$\mathcal{Y}$: $y_2 \longrightarrow y_3$

Z: $x_1 \longrightarrow x_3$

可见当 $A=\{x_1, x_3, x_4\}$时 $N(A)=B=\{y_2, y_3\}$，图不存在完全匹配

4．修改标号，求新的等价子图 G_l

① $a = \min\limits_{\substack{x_i \in A \\ y_j \notin N_{G_l}(A)}} \{l(x_i)+l(y_j)-w_{ij}\}$

$= \min\limits_{\substack{i=1,3,4 \\ j=1,4,5}} \{l(xi)+l(y_j)-w_{ij}\}$

$=\min\{(7-5),(7-6),(7-3),(6-4),(6-3),(6-0),(3-0)(3-0),(3-0)\}$

$=1$

② $l'(x_1)=l(x_1)-a=7-1=6$

$l'(x_3)=l(x_3)-a=6-1=5$

$l'(x_4)=l(x_4)-a=3-1=2$

$l'(y_2)=l(y_2)+a=0+1=1$

$l'(y_3)=l(y_3)+a=0+1=1$

其余结点的标号不变。

③ 将新的标号标在矩阵相应的右侧和下方，并且把对应 G_l 的矩阵元素用黑体字标出，得到下面矩阵，与之对应的新的等价子图 G_l 如图 5.25 所示（图中匹配边仍用粗线表示）

$$W=\begin{matrix} & y_1 & y_2 & y_3 & y_4 & y_5 & \\ x_1 & 5 & \mathbf{7} & \mathbf{7} & \mathbf{6} & 3 & 6 \\ x_2 & \mathbf{4} & 4 & 0 & \mathbf{4} & \mathbf{4} & 4 \\ x_3 & 4 & \mathbf{6} & \mathbf{6} & 3 & 0 & 5 \\ x_4 & 0 & \mathbf{3} & \mathbf{3} & 0 & 0 & 2 \\ x_5 & 3 & 4 & 3 & \mathbf{5} & \mathbf{5} & 5 \\ & 0 & 1 & 1 & 0 & 0 & \end{matrix}$$

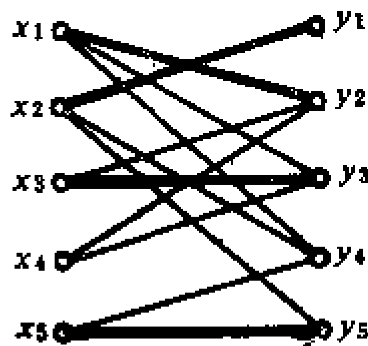


图 5.25

5．因 $y_4 \in N(A)-B$ 是M非饱和点，因此存在一条从 x_4 到 y_4 的可增广路

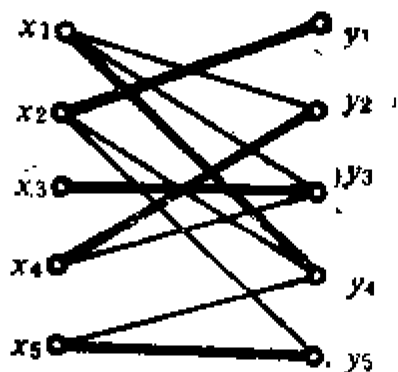


图 5.26

$$P=x_4y_2x_1y_4$$

$$E(P)=\{x_4y_2,\ y_2x_1,\ x_1y_4\}$$

因而得到新的匹配

$$M=\{x_1y_4,\ x_2y_1,\ x_3y_3,\ x_4y_2,\ x_5y_5\}$$

如图 5.26 中粗线所示。

6．至此，结点已全部饱和，G_l 已达到完全匹配，也是原带权完全偶图的最优匹配，总的最大利润为

$$W=6+4+6+3+5=24$$

这里要说明一点，上面给出的算法是在带权完全偶图的基础上进行的，如果提出的问

题所得到的不是完全偶图，我们可以增添一些权值为零的边使它成为完全偶图，例 5.7 的利润矩阵中元素为零所对应的边，就相当于添加边。此外，如果偶图的两个互补结点子集 V_1 和 V_2 的结点数目不等，我们也可以增添虚拟结点以满足 $|V_1|=|V_2|$，连接虚拟结点的边权值亦都为零。这时求出的最优匹配，是原图的最大权匹配，不一定是原图的最大基数匹配或完全匹配。现举两个例子说明如下。

例 5.8 求图 5.21(a)的带权偶图的最大权匹配。

解：原图不是完全偶图，为此我们增添一些权值为零的边使之成为完全偶图，得到相应矩阵如下，并将初始标号标在矩阵的相应位置上，于是得到对应的等价子图 G_l 如图 5.27。

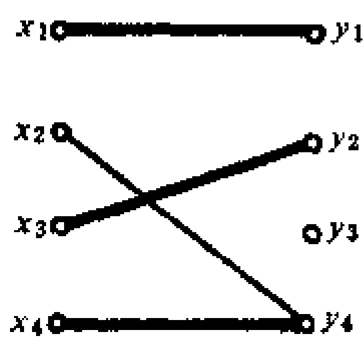


图 5.27

$$W=\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ \\ \end{array}\begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \\ \left(\begin{array}{cccc} \mathbf{3} & 2 & 0 & 0 \\ 3 & 0 & 4 & \mathbf{8} \\ 0 & \mathbf{5} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{2} \end{array}\right. \\ 0 & 0 & 0 & 0 \end{array}\left.\begin{array}{c} 3 \\ 8 \\ 5 \\ 2 \end{array}\right)$$

在图 G_l 中，取一初始匹配（图中粗线所示）

$$M=\{x_1y_1,\ x_3y_2,\ x_4y_4\}$$

并按以下过程寻找 M 可增广路

A：$x_2 \longrightarrow x_2, x_4$

$N(A)$：$y_4 \longrightarrow y_4$

B：$\phi \longrightarrow y_4$

Y：y_4

Z：x_4

此时 $A=\{x_2, x_4\}$，$N(A)=B$，图不存在完全匹配

修改标号，求出新的等价子图

$$a=\min_{\substack{i=2,4 \\ j=1,2,3}}\{l(x_i)+l(y_j)-w_{ij}\}=2$$

求出结点新的标号并标在下面矩阵的相应位置上，由此得出新的等价子图 G_l 如图 5.28。

$$W=\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ \\ \end{array}\begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \\ \left(\begin{array}{cccc} \mathbf{3} & 2 & 0 & 0 \\ 3 & 0 & 4 & \mathbf{8} \\ 0 & \mathbf{5} & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{2} \end{array}\right. \\ 0 & 0 & 0 & 2 \end{array}\left.\begin{array}{c} 3 \\ 6 \\ 5 \\ 0 \end{array}\right)$$

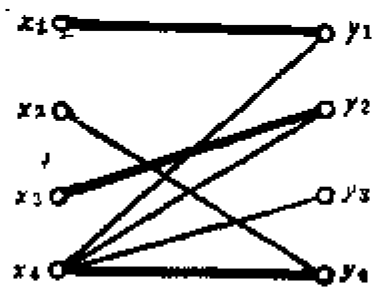


图 5.28

再按以下过程寻找 M 可增广路：

A：$x_2, x_4 \longrightarrow x_1, x_2, x_4 \longrightarrow x_1, x_2, x_3, x_4$

$N(A)$：$y_1, y_2, y_3, y_4 \longrightarrow y_1, y_2, y_3, y_4 \longrightarrow y_1, y_2, y_3, y_4$

B：$y_4 \longrightarrow y_1, y_4 \longrightarrow y_1, y_2, y_4$

Y：$y_1 \longrightarrow y_2 \longrightarrow y_3$

Z：$x_1 \longrightarrow x_3$

因 y_3 是M非饱和点，因而存在一条从 x_2 到 y_3 的可增广路

$$P = x_2, y_4, x_4, y_3$$

$$E(P) = \{x_2y_4, y_4x_4, x_4y_3\}$$

于是得到新的匹配

$$M = \{x_1y_1, x_2y_4, x_3y_2, x_4y_3\}$$

如图 5.29，它就是等价子图 G_l 的完全匹配，将此图与图 5.21(c)对照，可见它是原图的最大权匹配而不是最大基数匹配，因为边（x_4, y_3）是添加边。

例 5.9 求图 5.30 所示带权偶图的最大权匹配。

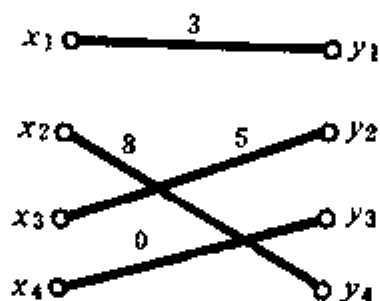


图 5.29

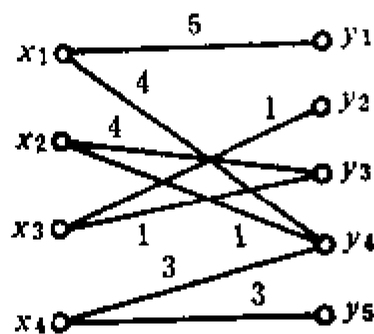


图 5.30

解：增加虚拟结点和添加权值为零的边，使原图成为完全偶图，得到相应矩阵如下，将初始标号标在矩阵的相应位置上，由标号得到等价子图 G_l 如图 5.31，取初始匹配：

$$W = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccc} y_1 & y_2 & y_3 & y_4 & y_5 \end{array} \\ \left[\begin{array}{ccccc} 0 & \mathbf{5} & 0 & 4 & 0 \\ 0 & 0 & \mathbf{4} & 1 & 0 \\ 0 & \mathbf{1} & \mathbf{1} & 0 & 0 \\ 1 & 0 & 0 & \mathbf{3} & \mathbf{3} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array}\right] \\ \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \end{array} \end{array} \begin{array}{c} \\ 5 \\ 4 \\ 1 \\ 3 \\ 0 \\ \\ \end{array}$$

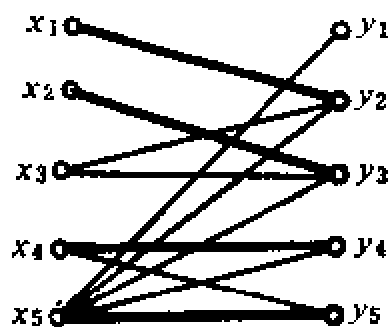


图 5.31

$$M = \{x_1y_2,\ x_2y_3,\ x_4y_4,\ x_5y_5\}$$

并按以下过程找M可增广路

A：$x_3 \longrightarrow x_3$，$x_1 \longrightarrow x_3$，x_1，x_2

$N(A)$：y_2，$y_3 \longrightarrow y_2$，$y_3 \longrightarrow y_2$，y_3

B：$\phi \longrightarrow y_2 \longrightarrow y_2$，$y_3$

Y：$y_2 \longrightarrow y_3$

z：$x_1 \longrightarrow x_2$

此时 $N(A) = B$，图不存在完全匹配。

修改标号，求出新的等价子图

$$a = \min_{\substack{i=1,2,3 \\ j=1,4,5}} \{l(x_i) + l(y_j) - w_{ij}\} = 1$$

求出结点新标号并标在下面矩阵相应位置上，由此得出新的等价子图 G_l 如图 5.32。

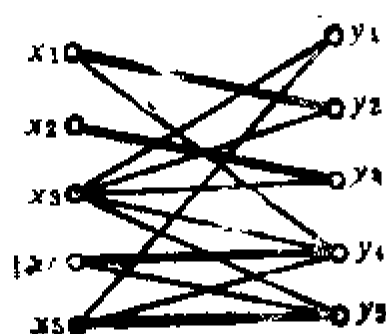


图 5.32

$$
\begin{array}{c|ccccc|c}
 & y_1 & y_2 & y_3 & y_4 & y_5 & \\
\end{array}
$$

$$
W=\begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix}\begin{pmatrix} 0 & \mathbf{5} & 0 & \mathbf{4} & 1 \\ 0 & 0 & \mathbf{4} & 1 & 0 \\ \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 1 & 0 & 0 & \mathbf{3} & \mathbf{3} \\ \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} \end{pmatrix}\begin{matrix} 4 \\ 3 \\ 0 \\ 3 \\ 0 \end{matrix}
$$

$$
\begin{matrix} 0 & 1 & 1 & 0 & 0 \end{matrix}
$$

在新的图 G_l 中寻找M可增广路，过程如下：

A：x_1，x_2，x_3

$N(A)$：y_1，y_2，y_3，y_4，y_5

B：y_2，y_3

y：y_1

y_1 是M非饱和点，因此存在从 x_3 到 y_1 的可增路

$$P=x_3,\ y_1,\ E(P)=\{x_3,\ y_1\}$$

于是得到新的匹配

$$M=\{x_1y_2,\ x_2y_3,\ x_3y_1,\ x_4y_4,\ x_5y_5\}$$

如图 5.33，而

$$W=5+4+3=12$$

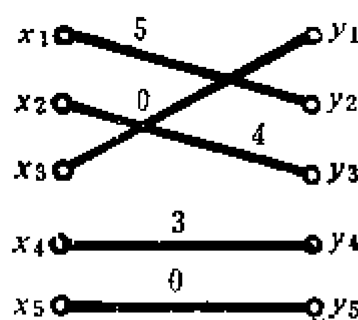


图 5.33

它即是原图的最大权匹配而不是最大基数匹配或完全匹配，因为边（x_3，y_1），（x_5，y_5）以及结点 x_5 都是原图所没有的。

有时要求匹配具有最小的权，（例如当边权表示成本或耗费的时间时），此时仍可采用上述算法，只要取一大于题给最大边权的值 $w_{\max}$，令

$$\bar{w}_{ij}=w_{\max}-w_{ij}$$

作出矩阵

$$\overline{W}=(\bar{w}_{ij})_{n\times n}$$

则求$\overline{W}$的最大权匹配即等于原题的最小权匹配。

在库恩—曼克莱斯算法中，计算 G_l 所需的次数显然是 $O(n^2)$级的，而在找出M可增广路之前，需要经过的循环次数最多为$|V_1|$次，并且在找出最优匹配之前初始匹配最多可能扩张$|V_1|$次，因此，这一算法不失为一个较好的算法。

偶图的匹配问题，也可转化为网络流问题求解，将在第九章讨论。

§5.5　一般图的最大基数匹配

这一节我们讨论一般图即非偶图的最大基数匹配问题。首先应该说明，无论是偶图还是一般图，定理 5.2 是普遍成立的，因此，通过寻找M可增广路以增加匹配的边数，从而获得最大基数匹配，仍然是我们求一般图的最大基数匹配的途径。不过，由于不是偶图，一般图一定含有次数为奇数的简单回路，它的边数一般可记作 $2K+1$，当回路中匹配的边

数比较稀疏时，用求偶图完全匹配的算法尚可进行，当回路匹配的边数达到最大（K条）时，可以验证，再用偶图的算法就不行了。我们把匹配边数达到最大时的奇次回路称为"花苞"（Blossom）。因此，求一般图最大基数匹配的算法中需要解决的关键问题，就是怎样检测存在花苞以及怎样处理花苞。

先讨论怎样检测花苞的存在。

回顾我们在寻找M可增广路时，一定是先从一M非饱和点v_0出发，构造M交替路，事实上这一过程的结果，我们将得到一棵以v_0为根的树，称为交替树，如图5.34所示。当存在某一树叶是M非饱和点时，则从树根到这片树叶的唯一路径，就是M可增广路。如果所有树叶都是M饱和点，则不存在以v_0为起点的可增广路，称这样的交替树为匈牙利树（Hungarian Tree）。

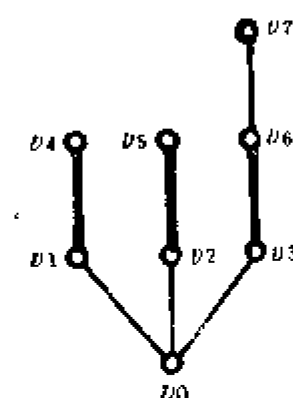


图 5.34

现在，我们把树的每条路径上的结点分为两类，即外部点和内部点，它们沿路径交替出现，并规定树根v_0为外部点，因此在图5.34中v_1，v_2，v_3，v_7都是内部点，而v_4,v_5,v_6都是外部点。这样标记以后即可得出如果树叶是内部点，则存在M可增广路。

定理 5.6 在构造M交替树的过程中，当且仅当两个外部点与同一条匹配边关联时，存在花苞。

证： 设在构造M交替树过程中出现花苞，如图5.35,以v_0为树根构造M交替树,由于存在奇次（$2K+1$）回路，当回路中匹配边数达到最大时出现花苞，此时回路上的所有结点均可标为外部点，这是因为如果从u_0沿着u_1,u_2，u_3的路径方向去标，则u_0，u_2，…，u_{2j-2}，u_{2j}，u_{2j+2}，…，u_{2K-2}，u_{2K}都是外部点，如果从u_0沿着u_{2K}，u_{2K-1}，u_{2K-2}的路径方向去标，则u_1，u_3，…，u_{2j-1}，u_{2j+1}，…，u_{2K-1}也都是外部点，即匹配边的两个端点都是外部点。

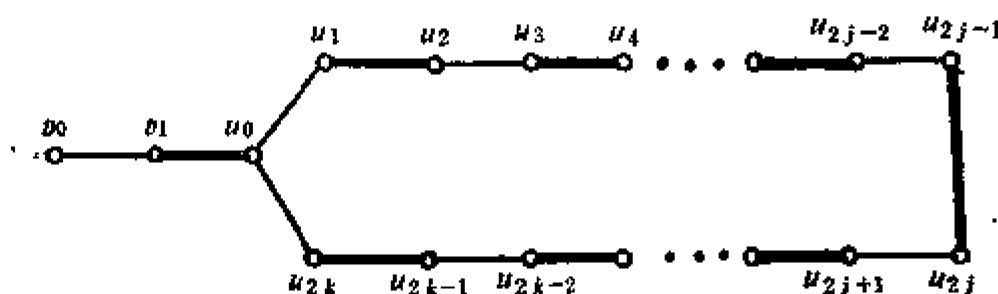


图 5.35

反之，设在标记过程中，一条匹配边的两个端点都是外部点，如图5.35中匹配边（u_{2j-1}，u_{2j}）的两个端点u_{2j-1}和u_{2j}都是外部点，说明在标记过程中匹配边（u_{2j-1}，u_{2j}）被使用了两次，现在从外部点v_{2j-1}沿着交替树生长的反方向回溯到树根v_0,路径为u_{2j-1}，u_{2j}，u_{2j+1}，…，u_{2K-1}，u_{2K}，u_0，v_1，v_0，显然路径的长度为偶数，同理，从外部点u_{2j}沿u_{2j-1}，u_{2j-2}，…，u_2，u_1，u_0，v_1回溯到v_0，路径的长度亦为偶数，回溯时不管这两条路径是在树根v_9处相交，还是在交替树的任一节点处相交（如图中是在节点u_0处相交），形成的回路其长度一定是奇数，由于回路中匹配的边数已达最大，因而存在花苞。 ▌

定理的证明也为我们提供了寻找花苞的方法，即从一条匹配边的两个外部点开始沿交

替树的生长方向回溯，当交于某一点 u_0 时，构成的回路即是花苞。称交点 u_0 为花苞的基结点，可以看到，在花苞内部除基结点外，其余的结点都是M饱和点。

当构造交替树过程中出现花苞时，怎样处理呢？由于花苞内部匹配边数已达最大，不可能再增加，自然会产生一种想法，就是把花苞看成图的一个点，不再探索它内部的匹配，只寻找图其余部分的可增广路，以求匹配边数的增加，这样就把复杂的图简化为不含花苞的图，也就可以采用前面的计算方法了。

所谓收缩花苞，就是把花苞当作一个结点，或者说把花苞的所有结点用一虚拟的结点代替，称之为伪点（pseudo—vertices），花苞外部所有与花苞中结点关联的边，都改为连接到伪点上。

如图 5.36 表示花苞的收缩过程，图中的粗线表示已给出的匹配边，图（a）从 0 点

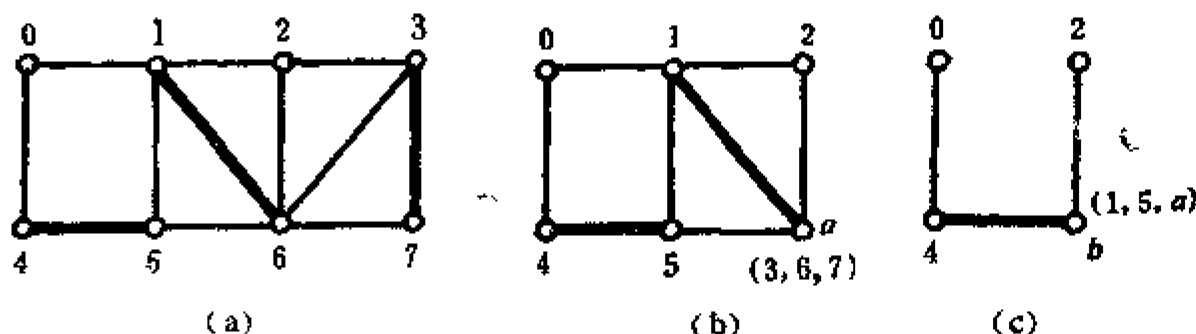


图 5.36

开始对各结点进行标记，结果发现匹配边（3，7）的两个端点都是外部点，于是得到花苞 $C_1=(3, 6, 7)$，将花苞收缩成一个结点 a，得到图（b），在图中又可找到花苞 $C_2=(5, 1, a)$，将它收缩成一个点 b，得到图（c）。由此可见，在收缩花苞的过程中，一个花苞内部可以包含一个或多个花苞。

当图不存在花苞即不存在奇次回路时，就可用前一节的算法求最大基数匹配，然后将花苞逐个打开（即将伪点恢复为原来的奇次回路），再求出各花苞内部的匹配边，最后即得原图的最大基数匹配。

花苞内部匹配边的选取，有两种情况：

（1）如果花苞收缩成的伪点是所在图的M饱和点，则打开花苞后就以它的基结点为初始点，按交替路径的生长过程取匹配边，即可得到奇次回路的最多匹配边。

（2）如果伪点不是所在的图的M饱和点，则打开花苞后可以花苞内任一结点为初始结点，按交替路径的生长过程取匹配边，最后也得到奇次回路内的最多匹配边。

以图 5.36（c）为例，这时存在可增广路，得到新的匹配如图 5.37(a)，将图(a)

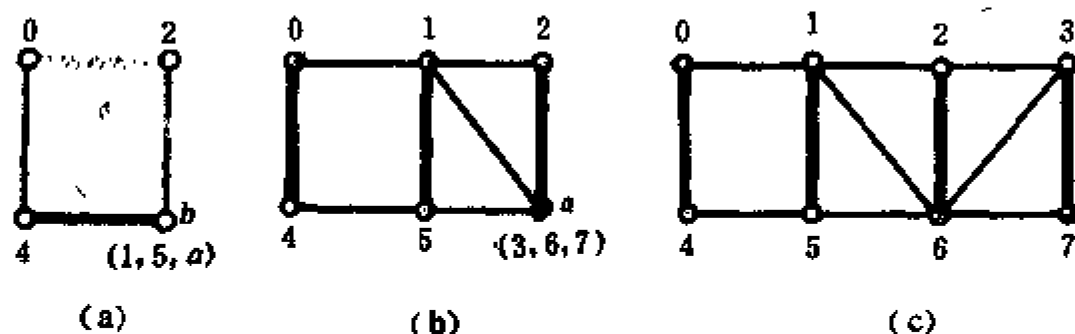


图 5. 37

中的花苞（伪点 b）打开，得到图（b），在奇次回路 $C_2=(1, 5, a)$ 中取匹配边 (1,5)，再将花苞（伪点 a）打开得到图（c），在奇次回路 $C_1=(3, 6, 7)$ 中取匹配边（3，7），

图（c）的匹配，即是图5.36（a）的一个最大基数匹配。

应用上述过程求一般图的最大基数匹配，它的正确性由下面的定理得到证明。

定理 5.7 设M是图G的一个匹配，若在寻找以M非饱和点v_0为始点的可增广路中，出现花苞B，则G中存在M可增广路的充要条件是花苞收缩以后得到的图G_b中亦存在M_b的可增广路。（仍以v_0为始点，M_b表示B收缩后余下的匹配）

证：设G_b存在以v_0为始点的增广路P，若P不穿过伪点v_b，则P本身就是G的一条可增广路。若P穿过v_b，如图5.38（a），则与v_b关联的两条边必然一条是匹配边，一条是非匹配边，不妨设(v_i, v_b)是匹配边，则在花苞中与这条匹配边关联的结点必然是基结点u_0，由于花苞中除u_0外其余的点都是M饱和点，故非匹配边(v_b, v_j)无论与花苞中任一点关联，都是M饱和点，不妨设该点为u_i，如图（b）所示，则在G中存在从v_0到v_p的一条路径为

$$P' = v_0, \cdots v_i, u_0, u_1, \cdots u_{i-1}, u_i, v_j, \cdots v_p$$

相当于在路径P中插入一段

$$P'' = u_0, u_1, \cdots, u_{i-1}, u_i$$

这段路径中匹配和非匹配边的数目是相等的，因此若P是可增广路，P'也一定是G中的可增广路。

反之，设G中存在一条以v_0为始点的M可增广路P，如果P在花苞收缩后并未穿过伪点v_b，则P必然也是G_b中一条以v_0为始点的M_b可增广路。若P穿过伪点v_b，可分两种情况：

（1）P以饱和边进入花苞而以非饱和边离开花苞。（如图5.38(b)），则在G_b中从v_0到v_p的路径也是M_b可增广路。（如图5.38(a)）。

（2）P以非饱和边进入花苞，这时又可分两种情况：

（a）P从花苞的基结点以饱和边离开花苞。这种情况与情况（1）类似，P经收缩后在G_b中仍是一条可增广路。

（b）P从花苞中的某一饱和点以非饱和边离开花苞，如图5.39，由于以v_0为根生成交替树时，若出现花苞必然存在花苞的基结点u_0，设从v_0经v_i到u_0的路径为

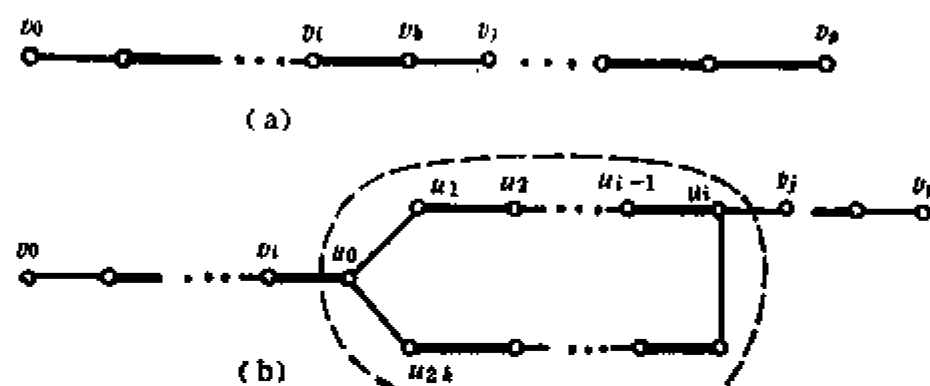


图 5.38

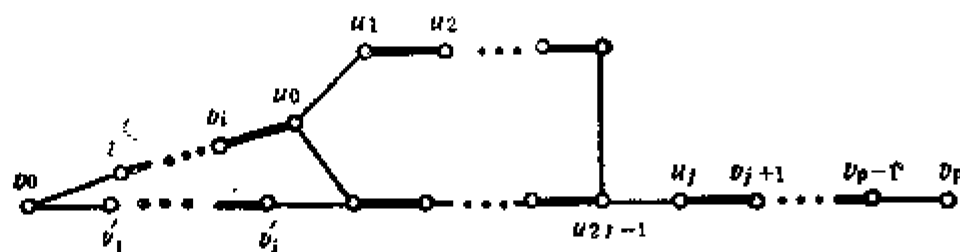


图 5.39

$$P'=v_0,\ v_1,\ \cdots,\ v_i,\ u_0$$

从花苞某一饱和点 u_{2i-1} 离开到 v_p 的路径为

$$P''=u_{2i-1},\ v_j,\ v_{j+1},\ \cdots v_{p-1},\ v_p$$

若 P' 与 P'' 不相交，当花苞收缩为伪点 v_b 后显然存在一条从 v_0 到 v_p 的路径

$$P'''=v_0,\ v_1,\ \cdots,\ v_i,\ v_b,\ v_j,\ v_{j+1},\ \cdots v_{p-1},\ v_p$$

它的长度是 P' 与 P'' 之和，显然 P''' 是 G_b 中从 v_0 到 v_p 的 M_b 可增广路。

如果路径 P' 与 P'' 相交，亦可证明（此处从略）当花苞收缩后，G_b 亦存在从 v_0 到 δ 的可增广路。 ∎

推论：设 M_b 是图 G 收缩花苞后成为图 G_b 时的最大基数匹配，则打开花苞后获得的匹配 M 是图 G 的最大基数匹配。

证：用反证法，设 M 不是 G 的最大基数匹配，则必然存在 M 非饱和点 v_0 的可增广路，由定理5.7，G_b 亦存在以 v_0 为始点的 M_b 可增广路，与 M_b 是 G_b 的最大基数匹配矛盾。 ∎

基于以上分析的思路，Edmonds于1965年提出了一般图最大基数匹配算法，算法给出求 M 可增广路（生成交替树）的过程及求最大基数匹配的算法步骤，分述如下：

一、求 M 可增广路过程 MAPS(G)

1．找一外部结点 $x\in T$ 及与 x 关联且未检测的边（x，y），找出后即将（x，y）记作已检测。如果不存在这样的边则转到 H。

2．如果 y 是非饱和点且未标记，则将边（x，y）加入交替树 T 中，转到 A。

3．如果 y 已被标记为外部点，则将（x，y）加入交替树 T 中，转到 B。

4．如果 y 已被标记为内部点，则回到1。

5．（不满足以上条件时，y 必然是未标记的饱和点）设（y，z）是以 y 为端点的 M 边，将（x，y）和（y，z）加入 T 中，并将 y 标记为内部点，z 标记为外部点，回到1。

二、最大基数匹配算法

1．选出 G_0 的任意一个初始匹配 M（或 $M\leftarrow\phi$），记所有 M 非饱和点为未检测点。

2．选择任一未检测的非饱和点 v，$T\leftarrow v$，并记 v 为外部点。$i\leftarrow 0$，如无则转7。

L：3．（生成交替树）MAPS(G_i)

B：4．（存在花苞）收缩花苞，$i\leftarrow i+1$，得伪点 b_i 及图 G_i，记 b_i 为外部点，转 L。

H：5．（出现匈牙利树）记 v 为已检测点，转2。

A：6．（存在可增广路），找出可增广路 P

$$M\longleftarrow M\oplus E(P)$$

转2。

7．（打开花苞），花苞内匹配边的选取，按前述方法进行

8．输出 M，计算结束。

三、算法说明

本算法分主程序和过程两部分，由于构造 M 交替树以寻找可增广路在计算中需要反复进行，所以把这一段语句序列作为过程以缩短程序。计算时可从任意一初始匹配开始（作为计算机程序，常置 $M\leftarrow\phi$），然后选择一非饱和点 v 作为交替树的根，调用过程 MAPS（G_i），在生成交替树过程中将出现如下五种情况：

（1）结点y是未标记的非饱和点，这时存在可增广路，于是转向出口A。

（2）结点y是未标记的饱和点，则继续应用过程以生长交替树。

（3）结点y是内部点，则将边(x,y)作为已检测，但不加入树中，继续生成交替树。

（4）结点y是外部点，将边（x，y）加入树中，树必然出现奇次回路（花苞），转出口B。

（5）与x关联的边都已检测，表明已不存在以v为根的可增广路，即生成的树已是匈牙利树，必须以另一非饱和点为根生成交替树，故转出口H。

当所有的非饱和点均已检测后，图即不存在可增广路，最后一步是将收缩的花苞依次打开，给出花苞内的匹配边，即得到原图的最大基数匹配。

例 5.10 求图 5.40 中G的最大基数匹配。

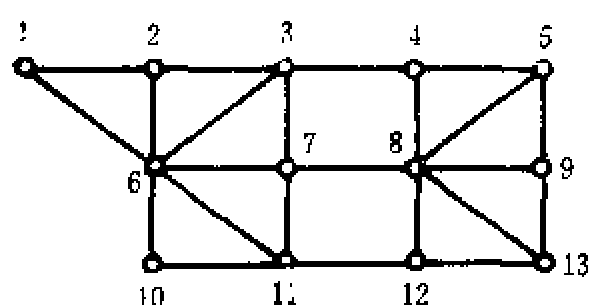


图 5.40

任取一初始匹配M，如图5.41(a)粗线所示，并取结点1为根生长交替树，得到一条交替路

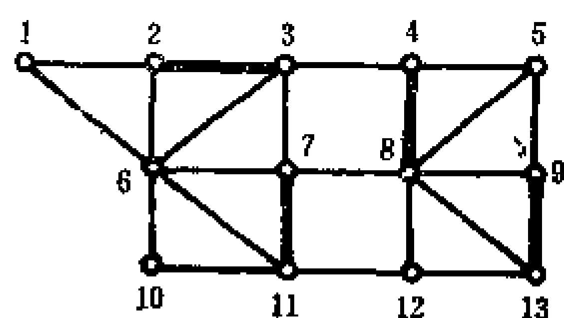
$$P=1,\ 2,\ 3,\ 4,\ 8,\ 9,\ 13$$

这时1，3，8，13均标为外部点。

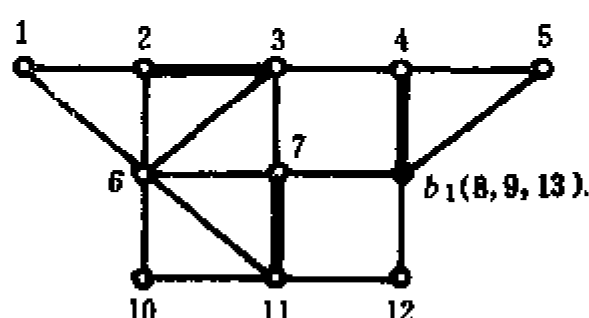
取外部点8，则边（8，13）为未检测边，且点13是外部点，将边(8,13)加入P中，出现花苞。

$$B_1=(8,\ 9,\ 13)$$

收缩花苞成为一伪点b_1，得到图5.41（b）



(a)



(b)

图 5.41

b_1为外部点，找到点5（未标记的非饱和点）得到可增广路

$$P=1,\ 2,\ 3,\ 4,\ b_1,\ 5$$

$$M\longleftarrow M\oplus E(P)$$

得到新的匹配如图5.42(a)中粗线所示。

选非饱和点6为根生长交替路

$$P=6,\ 3,\ 4,\ 5,\ b_1$$

4和b_1为外部点，将边（4，b_1）加入P中，出现花苞

$$B_2=(4,\ 5,\ b_1)$$

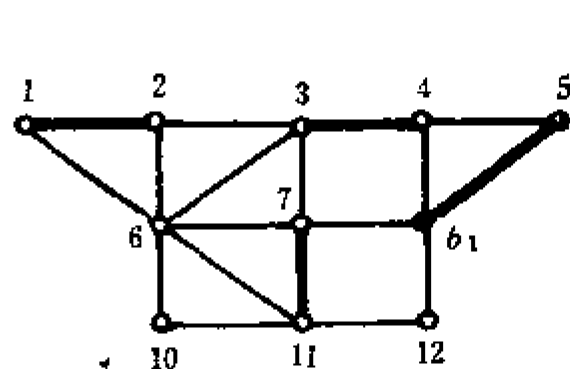
收缩花苞为一伪点b_2，得到图5.42(b)

这时点1已为M饱和，另选一M非饱和点6生成交替路

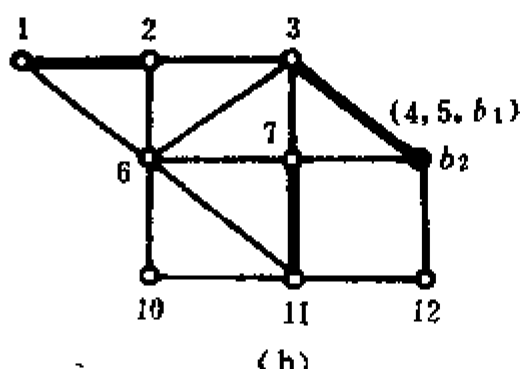
$$P=6,\ 7,\ 11,\ 12$$

P为一可增广路，令

$$M\longleftarrow M\oplus E(P)$$



(a)



(b)

图 5.42

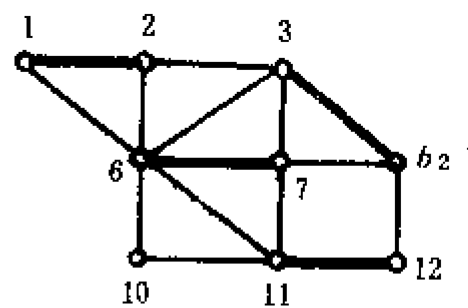
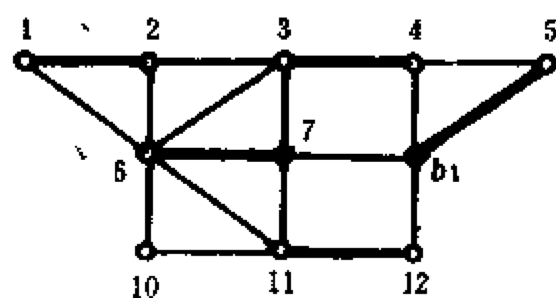


图 5.43

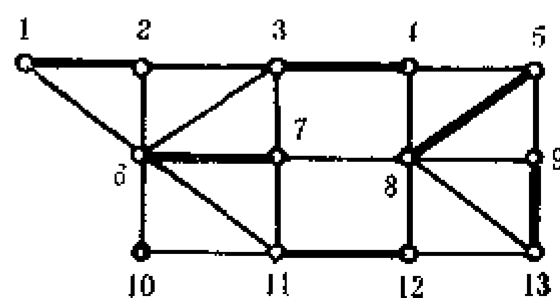
得到新的匹配如图5.43。

剩下最后一个非饱和点10，可以得到由它生成的树是一棵匈牙利树，此外，再无别的非饱和点可供选择，寻找可增广路至此结束。

最后一步是打开花苞，首先打开 b_2，选花苞内的匹配边，如图 5.44(a)，继而打开b_1，选取花苞内的匹配边，如图 5.44(b)，即得到原图的最大基数匹配。



(a)



(b)

图 5.44

一般来说，一个图的最大基数匹配不是唯一的，它取决于按怎样的次序选择非饱和点作为交替树的根结点，以及在交替树的生长过程中对外部点的选择。但无论怎样选择，最终得到匹配边的数目都是相同的。

可以看出，上面给出的求最大基数匹配的算法，它的计算复杂性是多项式量级的。计算时间主要耗费在寻找可增广路上，图有 n 个结点，因此增广次数不会超过 $0(n)$ 级，在寻找可增广路时可能还需要处理花苞，所花的时间也不会超过 $0(|E|)$，因此总的计算复杂性是 $0(n|E|)$ 级。

如果图存在完美匹配，那么用上述算法得到的就是图的完美匹配。显然图必须具有偶数个结点才可能有完美匹配，但这只是必要条件，并非充分条件。1947 年，Tutte 给出了判别图具有完美匹配的充要条件，下面不加证明地引出这一定理。

一个图的连通分支根据它具有奇数个或偶数个结点而分别称为奇分支或偶分支，我们用$\eta(G)$ 表示 G 的奇分支的个数。

定理 5.8 图 $G=(V, E)$有完美匹配的充要条件是，对所有的$S\subset V$，恒有

$$\eta(G-S)\leqslant|S| \tag{5.5}$$

§5.6 一般图的最大权匹配

这一节介绍由艾德蒙茨和约翰逊（Edmonds & Johnson）于1970年提出的求解带权图$G=(V, E)$ 的最大权匹配的一个算法。算法的思路与求最大基数匹配一样，也是形成

一棵交替树，从中寻找带权的M可增广路。所谓带权M可增广路，是一条匹配边和非匹配边交替出现的简单路径，其中非匹配边权的总和大于匹配边权的总和。

由于最大权匹配考虑的是匹配边权的总和为最大值，因此带权M可增广路中匹配边的数目不一定比非匹配边的数目少，而有如下定义中的几种类型。

定义 5.9 在带权M可增广路P中，如果：

（1）非匹配边比匹配边多，则称P为强可增广路。

（2）非匹配边与匹配边的数目相等，则称P为中性可增广路。

（3）非匹配边少于匹配边，则称P为弱可增广路。

如图5.45表示三条带权M可增广路（图中匹配边用粗线表示），P_1是强可增广路，P是中性可增广路，P_3是弱可增广路。

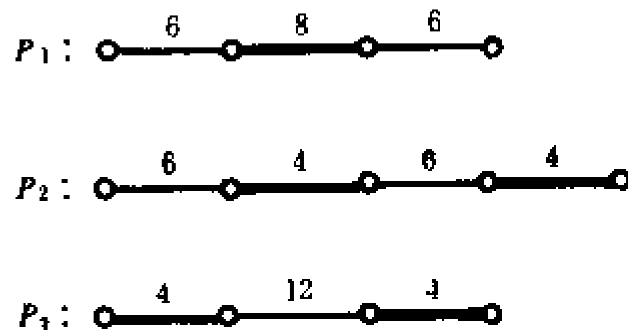


图 5.45

作为定理5.2的扩充，有如下定理。

定理 5.9 带权图G的匹配M是最大权匹配当且仅当G不含带权M可增广路。

证： 如果图G含有带权M可增广路P，则可构造一新的匹配$M'=M\oplus E(P)$，显然M'的权大于M的权，因而M就不是G的最大权匹配。

反之，如果M不是最大权匹配，我们将证明G必含有带权M可增广路。设M'是权比M更大的一个匹配，即$w(M')>w(M)$，令$G'=(V, M\oplus M')$，则G'的任一结点最多关联M'的一条边或最多关联M的一条边，即结点的次数最多是2，因此G'的每个连通分支是一条M和M'的边交替出现的路径（也可能是偶次回路），由于$w(M')>w(M)$，所以G'中至少有一分支，其中M'边权之和超过M边权之和，这一分支就是带权M可增广路。 ▮

本算法是应用线性规划中原始对偶松紧条件来设计求解图的最大权匹配的。算法中采用两个对偶变量y_v和Z_k，在算法的每次迭代时，通过调整量δ来调整对偶变量的值从而改变匹配关系，最后得到的匹配即是最大权匹配。限于篇幅，下面给出算法、算法说明及两个具体应用的例子，有关线性规划的知识及算法证明，不再介绍。

一、最大权匹配算法

1. $M\leftarrow\phi$，原图记为$G=(V,E)$

2. 对所有的$v\in V$

$$y_v=\frac{1}{2}\max\ \{w(v_i,v_j)\mid(v_i,v_j)\in E\}$$

$$Z_k=0$$

3. $E^*=\ \{(u,v)\mid y_u+y_v+\sum\limits_{(u,v)\in R_k}Z_k=w(u,v)\}$

$V^*=\ \{u\mid u$ 与 E^*的边关联$\}$

$G^*=(V^*,E^*)$

C：4 在V中任选一$y_v>0$且未检测的非饱和点v，$T\rightarrow v$，标v为外部点。

如果$v\notin V^*$，转H

如果找不到这样的点，转L。

M：5. MAPS(G^*)

B：6．得到花苞 R_k，将它收缩为一伪点 v_B，并标记为外部点、转M。

H：7．HUT

A：8．得到强带权增广路，将路径中匹配边和非匹配边交换。
去掉所有结点的标记，转C。

L：9．打开花苞，以相反次序将每一个伪点打开(即最后一个形成的伪点最先打开)，在所得的奇次回路中产生一个最大基数匹配。最后得到的匹配，即是G的最大权匹配。

二、δ调整量的调整过程 DEV

1．当花苞R_K收缩为伪点 v_B，且 v_B 标为内部点时，取

$$\delta_1=\frac{1}{2}\min_{R_k}\{Z_k\}$$

2．对 $v\in V$ 并标为外部点的结点，取

$$\delta_2=\min_{v}\{y_v\}$$

3．对 V 中所有标为外部点的 u 及未标记的结点 v，取

$$\delta_3=\min_{(u,v)}\{y_u+y_v-w(u,v)\}$$

4．对 u，v 都是外部点且不在一个伪点内时，取

$$\delta_4=\frac{1}{2}\min_{(u,v)}\{y_u+y_v-w(u,v)\}$$

5．取调整量

$$\delta=\min\{\delta_1,\ \delta_2,\ \delta_3,\ \delta_4\}$$

三、对偶变量调整过程 DVC

1．对所有外部点 u（包括标记为外部点的伪点中的结点）

$$y_u\longleftarrow y_u-\delta$$

2．对所有内部点 u（包括标记为内部点的伪点中的结点）

$$y_u\longleftarrow y_u-\delta$$

3．花苞 R_k 的伪点标为外部点时

$$Z_k\longleftarrow Z_k+2\delta$$

4．花苞 R_k 的伪点标为内部点时

$$Z_k\longleftarrow Z_k-2\delta$$

四、匈牙利树处理过程 HUT

1．DEV

2．DVC

3．如果 $\delta=\delta_1$，则存在某一标内部点的伪点其Z变量值为零，将这一伪点打开使回复为原来奇次回路，转M

4．如果 $\delta=\delta_2$ 且树根 v 的对偶变量 y_v 不为零，则存在从树根 v 到 y 变量为零的结点的一条中性可增广路P，将P中匹配边与非匹配边交换。

5．如果 $\delta=\delta_3$ 或 $\delta=\delta_4$，则将确定 δ_3 或 δ_4 的边（u，v）加入 E^*

6．如果 $\delta=\delta_2$，则去掉所有外部和内部点标记。转C。

7．如果 $\delta=\delta_1$ 或 $\delta=\delta_3$ 或 $\delta=\delta_4$，则转M。

算法说明：

1．求图 $G=(V, E)$ 的最大权匹配必须从匹配为空集开始，然后给定对偶变量 y_v 和 Z_k 的两个初值，每一结点 v 都有一个对偶变量 y_v，Z_k 是在发现花苞时出现的，它们的值在计算过程中得到调整。

2．根据 y_v 和 Z_k 的初始值，求出 E 中满足条件的边集 E^*，V^* 是 E^* 中边的端点集合，所以 $G^*=(V^*, E^*)$ 是 G 的一个子图。

3．求可增广路过程 MAPS(G^*)即交替树的生长过程，和求最大基数匹配的 $MAPS(G)$ 过程是相同的，但要注意，这里是以图 G^* 为对象，即在边集 E^* 中进行交替树时形成的，因此，当我们在 G 中取的非饱和点 v 不与 E^* 中的边关联时（即 $v\notin V^*$），它不可能在 E^* 中进行标记，即本身构成了一棵匈牙利树，所以都要转向 H，作匈牙利树的处理。只有当 $v\in V^*$ 时，才进行 $MAPS(G^*)$ 过程，和最大基数匹配算法一样，它有三个出口，当找到可增广路时，转向出口 A，找到奇次回路时，转向出口 B，找到的是匈牙利树时，转向出口 H。

4．随着对偶变量的调整，边集 E^* 的边不断增加，最后 $E^*=E$，交替树的生成在 E 中进行，最后得到的最大基数匹配，即是图的最大权匹配。

5．在算法进行中，将需要进行花苞的收缩或伪点的打开，一个花苞可能又包含多个花苞，算法中对花苞的标记，都是对最外层花苞而言的。

6．以上算法将 δ 量的调整、对偶变量的调整以及对匈牙利树的处理作为三个过程（加上交替树的生成共四个过程）与主程序分开，目的在于使程序更为简洁清晰，便于阅读。

这一算法的时间复杂性也是多项式级的，为 $0(n^3)$。

例 5.11 设某项工作必须两人一组配合完成，记 $w(v_i, v_j)$ 为 v_i 与 v_j 配合工作时所取得的效益，图 5.46 表示 a，b，c，d 四个人，边权代表边的端点对应的两个人合作时取得的效益，问应如何安排使所取得的总效益最大。

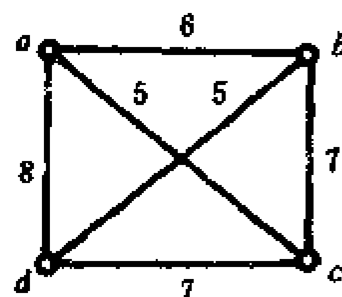


图 5.46

解： 这是一个求最大权匹配问题。应用上述算法将每次迭代过程各量的变化列成表如表 5.1 并分述如下：

对所有的 $v\in V$，令

$$y_v=\frac{1}{2}\max\{5, 5, 6, 7, 7, 8\}=4$$

$$Z_k=0$$

得 $$E^*\{(a, d)\}$$

C：一、检测点 a，标记为外部点。

MAPS(G^*)：E^* 中只有边 (a, d)，d 是非饱和点且未标记，故

$$T\longleftarrow \underset{外}{a} \longrightarrow d$$

转出口 A

A：(a, d) 是一强增广路，将 (a, d) 加入 M 中。去掉所有结点标记，转 C

C：二、检测点 b，因 $b\notin V^*$，转 H

HUT：DEV：$\delta_1=\infty$，$\delta_2=\min\{y_b\}=4$，

表 5.1

	y_a	y_b	y_c	y_d	Z_k	E^*	M
起 始	4	4	4	4	0	(a,d)	ϕ
检测 a	4	4	4	4	0		(a,d)
检测 b	4	3	4	4	0	(a,d), (b,c)	(a,d), (b,c)

$$\delta_3=\min\{4+4-6,\ 4+4-5,\ 4+4-7\}=1$$

$$\delta_4=\infty$$

$$\delta=\min\{\delta_1,\ \delta_2,\ \delta_3,\ \delta_4\}=1$$

DVC: $y_b=4-1=3$

因 $\delta=\delta_3$，将边 $(b,\ c)$ 加入 E^* 中，转M。

MAPS(G^*)：检查边 $(b,\ c)$，c 点非饱和点且未标记，故

$$T\leftarrow \underset{外}{b} — c$$

转出口A

A：$(b,\ c)$ 是一强增广路，将 $(b,\ c)$ 加入匹配M中，去掉所有结点标记，转C

C：三：因V 中已不存在非饱和点，转L

L：没有花苞须要打开，M即为最大权匹配

$M=\{(a,\ d),\ (b,\ c)\}$，$w(M)=8+7=15$。

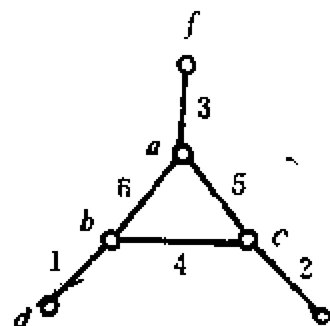


图 5.47

例 5.12 求图5.47的带权图的最大权匹配。

解： 令所有 $v\in V$

$$y_v=\frac{1}{2}\max\{1,\ 2,\ 3,\ 4,\ 5,\ 6\}$$

$$=3$$

$$Z_k=0$$

则 $$E^*=\{(a,\ b)\}$$

迭代过程中各量的变化如表 5.2 所示，并分述如下。

C：一、检测点 a，标 a 为外部点，因 $a\in V^*$

MAPS(G^*)：E^*中只有边 $(a,\ b)$，且 b 是未标记的非饱和点，故

$$T\leftarrow \underset{外}{a} — b$$

转出口A

A： $(a,\ b)$ 是强增广路，将 $(a,\ b)$ 加入M中，去掉所有结点标记，转C

C：二、检测点C，因 $C\notin V^*$，转H

HUT：DEV：$\delta_1=\infty$，$\delta_2=\min\{y_c\}=3$

$$\delta_3=\min\ \{3+3-5,\ 3+3-4,\ 3+3-2\}=1$$

$$\delta_4=\infty$$

$$\delta=\min\{\delta_1,\ \delta_2,\ \delta_3,\ \delta_4\}=1$$

表 5.2

	y_a	y_b	y_c	y_d	y_e	y_f	Z_k	E^*	M
起 始	3	3	3	3	3	3	0	(a,b)	ϕ
检测 a	3	3	3	3	3	3	0		(a,b)
检测 c	3	3	2	3	3	3	0	$(a,b),(a,c)$	
	3.5	2.5	1.5	3	3	3	0	$(a,b),(a,c),(b,c)$	除去 (a,b)
	2	1	0	3	3	3	3	$(a,b),(a,c),(b,c)$	
检测 d	2	1	0	0	3	3	3	$(a,b),(a,c),(b,c)(d,b)$	
检测 e	2	1	0	0	2	3	3	$(a,b),(a,c),(b,c)(d,b),(e,c)$	(e,c)
检测 f	2	1	0	0	2	1	3	$(a,b),(a,c),(b,c)$ $(d,b),(e,c),(a,f)$	
	3	2	1	0	1	0	1	同上	

DVC: $y_c=3-1=2$

因 $\delta=\delta_3$，将边（a, c）加入 E* 中，转M

MAPS(G^*)：检查边（a, c），得

$$T \leftarrow - - c \text{——} a \text{——} b$$

外　内　外　　（粗线表匹配边）

两个外部点 a, b 无法继续标记，转 H

HUT: DEV: $\delta_1=\infty$, $\delta_2=\min\{y_b, y_c\}=2$

$$\delta_3=\min\{y_b+y_d-w(b,d),\ y_c+y_e-w(c,e)\}$$
$$=\min\{3+3-1,\ 2+3-2\}=3$$

$$\delta_4=\frac{1}{2}\min\{y_b+y_c-w(b,\ c)\}=0.5$$

$$\delta=\min\{\delta_1,\ \delta_2,\ \delta_3,\ \delta_4\}=0.5$$

DVC: $y_b=3-0.5=2.5$, $y_c=2-0.5=1.5$

$y_b=3+0.5=3.5$

因 $\delta=\delta_4$，将边（b, c）加入 E^*中，转M

MAPS(G^*)：检查边（b, c）得到花苞 $R_B=(a,\ b,\ c)$，转B。

B：收缩花苞，令 $v_B=(a,b,c)$ 并标记为外部点，将花苞内匹配边从M中除去，转M

MAPS(G^*)：因 G^*中只有一个伪点，它自身成为一匈牙利树，故转H

HUT: DEV: $\delta_1=\infty$, $\delta_2=\min\{y_a,\ y_b,\ y_c\}=1.5$

$$\delta_3=\min\{3.5+3-3,\ 2.5+3-1,\ 1.5+3-2\}$$
$$=2.5$$
$$\delta_4=\infty$$
$$\delta=\min\{\delta_1,\ \delta_2,\ \delta_3,\ \delta_4\}=1.5$$

DVC:

$$y_a=3.5-1.5=2$$
$$y_b=2.5-1.5=1$$

$$y_c=1.5-1.5=0$$

$$Z_k=0+2\times1.5=3$$

因 $\delta=\delta_2$，但 $y_c=0$，故去掉结点的标记，转C

C：三、检测点 d，因 $d\notin V^*$，故转H

HUT：DEV：$\delta_1=\infty$，$\delta_2=\min\{y_d\}=3$，

$\delta_3=\min\{3+1-1\}=3$，$\delta_4=\infty$.

$\delta=\min\{\delta_1,\delta_2,\delta_3,\delta_4\}=3$

DVC：$y_d=3-3=0$

因 $\delta=\delta_3$，故将边 (d,b) 加入 E^* 中

因 $\delta=\delta_2$ 但 $y_d=0$，故去掉结点标记，转C

C：四、检测点 e，因 $e\notin V^*$，故转H

HUT：DEV：$\delta_1=\infty$，$\delta_2=\min\{y_e\}=3$

$\delta_3=\min\{y_e+y_c-w(e,c)\}=1$

$\delta_4=\infty$

$\delta=\min\{\delta_1,\delta_2,\delta_3,\delta_4,\}=1$

DVC：$y_e=3-1=2$

因 $\delta=\delta_3$，将边 (e,c) 加入 E^* 中，转M

MAPS(G^*)：检查边 (e,c)，得

$$T\longleftarrow \underset{外}{e}\longrightarrow v_B$$

转出口 A

A：(e,c) 是一强增广路，将边 (e,c) 加入匹配 M 中，去掉所有标记，转C

C：五、检测点 f，因 $f\notin V^*$，转H

HUT：DEV：$\delta_1=\infty$，$\delta_2=\min\{y_f\}=3$

$\delta_3=\min\{y_f+y_a-w(a,f)\}=2$

$\delta_4=\infty$

$\delta=\min\{\delta_1,\delta_2,\delta_3,\delta_4\}=2$

DVC：$y_f=3-2=1$

因 $\delta=\delta_3$，将边 (a,f) 加入 E^* 中，转M

MAPS(G^*)：检查边 (a,f) 得

$$T\longleftarrow \underset{外}{f}\longrightarrow \underset{内}{v_B}\longrightarrow \underset{外}{e}$$

两个外部点 f，e 无法继续标记，转H

HUT：DEV：$\delta_1=\frac{1}{2}[Z_k]=1.5$

$\delta_2=\min\{y_f,y_e\}=\min\{1,2\}=1$

$\delta_3=\infty$，$\delta_4=\infty$

$\delta=\min\{\delta_1,\delta_2,\delta_3,\delta_4\}=1$

DVC：$y_f=1-1=0$

$y_e=2-1=1$

$$y_a=2+1=3,\ y_b=1+1=2,\ y_c=0+1=1$$

$$Z_k=3-2=1$$

因 $\delta=\delta_2$ 且 $y_f=0$，去掉结点的标记，转C。

C：六、已无未检测点，转L。

L：七、打开花苞 $v_B=(a,b,c)$，因 (c,e) 是匹配边，故取 (a,b) 为匹配边．得

$$M=\{(a,b,\ (c,e)\}$$

此即是图G的最大权匹配，总权值为

$$W(M)=6+2=8$$

习题与思考题

1．图 5.48 的图是否是偶图，如果是，求出它的互补结点子集。

2．证明：如果G是偶图，它有 n 个结点m条边，则 $m\leqslant n^2/4$。

3．证明：一棵树最多只有一个完美匹配。

4．试求 $K_{n,m}$中不同的完美匹配的个数。

5．对于图 5.49 的偶图，试证明

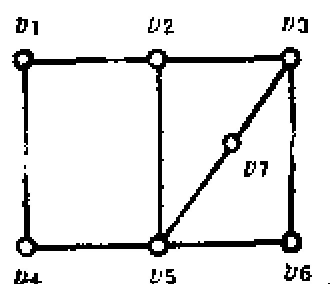


图 5.48

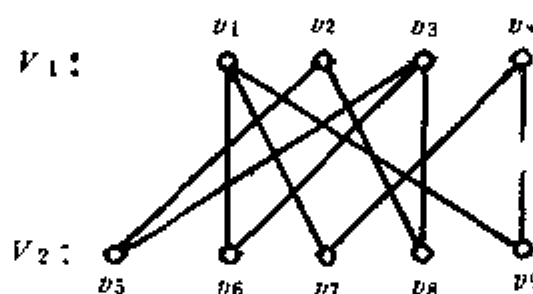


图 5.49

（1）它满足相异性条件。

（2）它满足 t 条件。

6．出席某次国际学术报告会的 6 个成员 a,b,c,d,e,f，他们的情况是：

a：会讲汉语、法语和日语。

b：会讲德语、日语和俄语

c：会讲英语和法语

d：会讲汉语和西班牙语

e：会讲英语和德语

f：会讲俄语和西班牙语

欲将此 6 人分为两个组，使得同一组中任何两个人不能互相交谈，而两组之间可以相互交谈，问应如何分组。

7．有四名教师：张明、王同、李林和赵丽，欲分派他们去教四门课：数学、物理、电工和图论，已知这四位教师的情况是：

张明：能教物理、电工

王同：能教数学和图论

李林：能教数学、物理和电工

赵丽：能教电工

问应如何分派他们的工作

8．一次舞会，共有 n 个男士和 n 个女士参加，已知每 2 个男士至少认识 2 个女士，而每个女士最多认识 2 个男士，问能否把男士和女士正好分成 n 对，使每一对男女都彼此认识。

9．某杂志公布了 7 个征求答案的题目，当从读者寄来的解答中挑选每道题的两个解答准备下期发表时，编辑发现所有14个挑选出来的解答恰好是 7 个读者提出的，而且他们每个人正好提供了两个答案。试证明：编辑可以怎样发表每道题的解答，使得在发表的解答中，这 7 个读者中的每一个人都有一个解答。

10．a，b，c，d，e，f 六个人组成一个小组检查五个单位的工作，若某单位和 b、c、d 三人有过工作联系，则用$\{b,c,d\}$表示，其余四个单位分别是$\{a, e, f\}$，$\{a, b, e, f\}$，$\{a, b, d, f\}$，$\{a, b, c\}$，若到一单位检查工作的人，必须是与该单位没发生过联系的人，问应如何安排。

11．设 $A=(a_{ij})_{n\times m}$ 是一个布尔矩阵，且 $m\leqslant n$，A 矩阵每行都有 k 个 1 元素，而每列 1 元素的个数不超过 k 个，试证：

$$A=P_1+P_2+\cdots+P_k$$

其中 P_i 也是一个 $n\times m$ 的布尔矩阵，每行有一个 1 元素，每列 1 元素的个数不超过一个。

12．已知用 x_1,x_2,x_3,x_4 四种原料制造 y_1，y_2，y_3，y_4 四种产品的成本如下面矩阵所示，问采用哪种方案可使成本最低（假定用原料制作某种产品就不能再用来制作其他产品）

$$E=\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array}\begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 99 & 6 & 59 & 73 \\ 79 & 15 & 93 & 87 \\ 67 & 93 & 13 & 81 \\ 16 & 79 & 86 & 26 \end{bmatrix} \end{array}$$

13．两个人在图 G 上做游戏，方法是交替地选择相异的结点 v_0，v_1，$v_2\cdots$，使得对每个 $i>0$，v_i 邻接于 v_{i-1}，最后一个结点的选择者得胜。试证明：每一个选点人有一个得胜策略当且仅当图 G 没有完美匹配。

14．对于图 5.50 所示的偶图，试构造

（1）最大基数匹配。

（2）最大权匹配。

15．对于图 5.51 所示的图，试构造

（1）最大基数匹配。

（2）最大权匹配。

16．最大基数匹配算法如何把已经错误地放入解中的一条边除去？最大权匹配如何解决这个问题呢？

17．试证明：完全图 k_{2n} 中不同的完美匹配的个数是

$$N=(2n)!/2^n\cdot n!$$

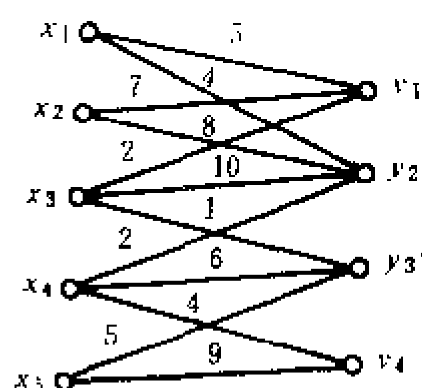


图 5.50

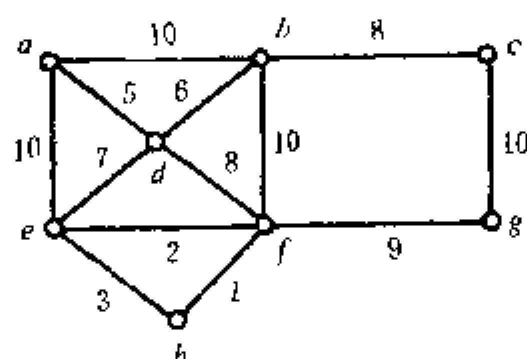


图 5.51

18．某机加车间拥有六台不同的钻床。某日来了五个需要钻孔的工件，在每台钻床上对每个工件进行加工所需的工时如下所示，求每一个工件都在不同机床上加工最优方法。

机床 \ 工件	A	B	C	D	E
1	5	7	6	4	9
2	8	10	3	4	7
3	6	11	5	4	7
4	5	8	7	3	9
5	3	6	4	2	7
6	3	7	5	3	7

19．联合国发起姐妹城市活动，城市双双地配对进行文化交流活动，今有10个新的城市申请参加，试求这些城市双双配对的最好方案以使所有姐妹城市之间的距离总和为最短，城市之间的距离如下。

从城市 \ 至城市	1	2	3	4	5	6	7	8	9	10
1	0	80	70	70	60	45	90	110	85	155
2		0	75	95	90	80	90	160	70	15
3			0	65	70	60	100	80	80	55
4				0	80	80	70	170	200	250
5					0	110	170	190	270	300
6						0	100	150	110	200
7							0	75	95	100
8								0	90	100
9									0	50

第六章 图的着色

§6.1 地图的着色

所谓图的着色，是指对图的每个结点（或边或平面图的面）指定一种颜色，使相邻的结点（边或面）不同色。

对图着色的研究，不仅有重大的理论意义和应用价值，而且也有悠久的历史。它的起源可以追溯到150年前，由对地图的着色而引起，即提出了这样一个问题：在一个平面上的任何地图，是否可以最多只用四种颜色就能使相邻的国家着不同的颜色。（两个国家相邻是指它们有一段公共的边界线，而不是仅有一个公共点）。

例如对图6.1(a)的平面图的面着色，由于面E与B，面C与D不相邻，图的着色可以不多于四种颜色，而对(b)图的着色，无论怎样安排，少于四种颜色都是不行的。

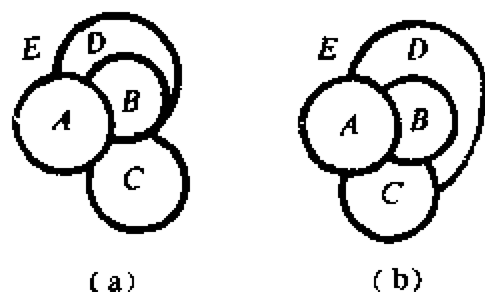


图 6.1

1852年，格思里（Guthrie）把地图的着色最多只须四种颜色的猜想告诉他的哥哥弗莱得里克（Frederick），哥解决不了转而请教他的老师德·摩根（D. Morgan）。这一命题从直观或实践中都表明是正确的，但要用数学给予严格的证明却遇到了很多困难，因而称为"四色猜想"。1878年，凯莱（Cayley）把它提到伦敦数学学会上，"四色猜想"公诸于世，引起了各国数学界的注意和兴趣。

1879年，肯普（Kempe）给出了这个猜想的许多个错误"证明"中的第一个"证明"，11年后，即1890年，希伍德（Heawood）发现了它的一个错误，但希伍德指出，如果把"四"换成"五"，这个猜想就对了，因而提出了"五色定理"，但是仅仅过了一年，赫夫特（Heffter）就指出希伍德在五色定理的论证中疏忽的几个错误并作了修正。1969年，奥尔（Ore）和斯坦普尔（Stemple）对少于40个国家的所有地图，证明了"四色猜想"的正确性。然而对任意多个国家，"四色猜想"的正确性仍然没有证明出来。一百多年来这一数学难题吸引着众多数学家和数学爱好者，有的甚至为它探索终身，虽然没有获得成功，但对这个问题的研究促进了图的可着色性的研究，取得了很多重要的成果，为今后继研究奠定了良好的基础，而且也导致了对图论其他一些领域的探讨，促进了图论的理论和应用的发展。

1976年，美国青年数学家阿佩尔（Appel）和黑肯（Hakan）首次应用计算机证明了这一难题，使"四色猜想"成为"四色定理"。他们从1976年1月至6月在三部计算机上用了1200小时，作了近百亿次逻辑判断才得出结果，除证法奇特之外，一些关系重大的计划也是由计算机实验来完成的，证明的正确性如果不借助计算机也无法检验，其复杂程度可想而知，限于篇幅，这里就不作介绍了。

由于五色定理在历史上对四色猜想的证明，起到一定的促进作用，也有助于我们对问

题的思考，下一节我们将对五色定理的证明作一介绍。

§6.2 五 色 定 理

任一平面图都有一个对偶图，平面图的面和它的对偶图的结点有着一一对应的关系，所以对平面图面的着色，可归化为对其对偶图点的着色，显然研究平面图点的着色要比研究面的着色更为方便，所以对地图面的着色都是归化为点的着色进行研究的。

定义 6.1 平面上一条连续的自身不相交且始点与终点相重合的曲线，称为约当（Jordan）曲线。

例如平面图的任一个面它的周界的各条边的并集构成一条约当曲线。

显然可见，若 J 是平面上的一条约当曲线，则 J 把平面分成两部份，一部份是有限面，称为 J 的内部，一部份是无限面，称为 J 的外部。由此得到著名的约当曲线定理如下

定理 6.1 若 J 是平面上的一条约当曲线，x 和 y 分别是 J 内部和外部上的任意两个点，则 x 和 y 的任何连线必与 J 至少有一交点。

定理的正确性在直观上是显然的，如图 6.2 所示。但它的严格证明却十分困难。这里从略。

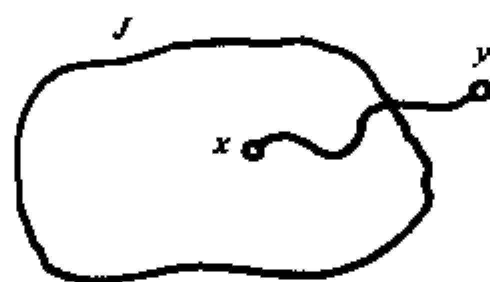


图6.2

定义 6.2 对图 $G=(V, E)$ 的结点着色，使任意两个相邻的结点都不同色，则称这种着色是正常的（或正常着色）。若正常着色时使用了 k 种颜色，则称图 G 是结点 k 可着色的。

定义 6.3 对图 G 的结点正常着色需要最少颜色的种数，称为图 G 的点色数，记作 $\varphi_v(G)$。

如果图 G 是结点 k 可着色的，显然有 $\varphi_v(G)\leqslant k$。以后讨论图的着色都指正常着色，因此正常二字可以略去，简称为图的着色。

定理 6.2 任意一个平面图都是结点 5 可着色的。

证：对图 G 的结点数用归纳法证明如下：

当结点数 $n\leqslant 5$ 时，命题显然成立。

设 $n-1$ 个结点时命题成立，下面证明 n 个结点时命题亦成立。

由于平行边和自环不影响点的色数，所以任意一个平面图都可当作简单平面图处理，由定理 4.2 的推论 2 知，一简单平面图至少有一结点的次数小于等于 5，设此结点为 v_0，将 v_0 以及与它关联的边从图中删去，得图 $G_0=G-\{v_0\}$ 则 G_0 有 $n-1$ 个结点，根据假设前提，G_0 是 5 可着色的。

现在将 v_0 及原来与它关联的边仍旧加到 G_0 上，图又恢复为 n 个结点的图 G，这时有两种可能：

（1）v_0 的次数小于 5，即 $deg(v_0)<5$。

在这种情况下，与 v_0 邻接的结点不超过 4 个（或者说最多有 4 个），即与 v_0 邻接的结点最多用了 4 种颜色，则 v_0 可以着与它的邻接点不同的颜色，而不会使着色数超过 5 种，因此图是 5 可着色的，命题得证。

（2）v_0 的次数等于 5，即 $deg(v_0)=5$。

不失一般性，假定与 v_0 邻接的 5 个结点已分别着上了 5 种不同的颜色：红、白、黄、蓝、黑，如图 6.3 所示。

现在看图 G 中 v_0 以外的其他点的情况，我们把所有着上红、黄色的结点集合称为红黄集，把所有着上黑、白色的结点集合称为黑白集，这时将 v_0 去掉后可能出现两种情况：

（a）去掉 v_0 后结点 v_1 与 v_3 分处在两个连通分支中，如图 6.4 所表示的那样，这时

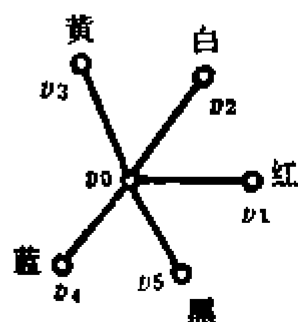


图 6.3

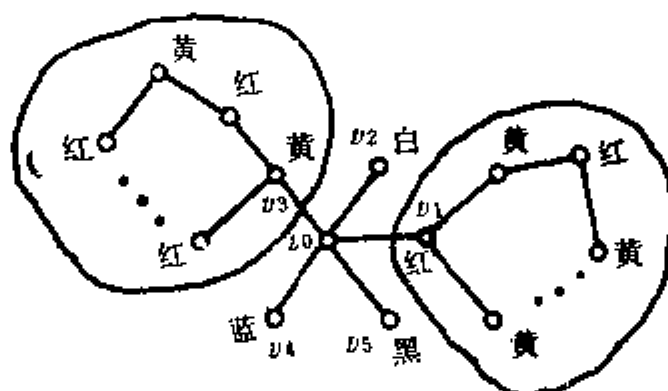


图 6.4

将 v_3 所处的连通块中的红黄色对换（即原来着黄色的点改着红色，原来着红色的点改着黄色），并不影响整个图的正常着色，于是 v_3 改着红色，则与 v_0 邻接的 5 个结点只着 4 种颜色：白、红、蓝、黑、红，因而 v_0 可着第 5 种颜色：黄色。于是图 G 是 5 可着色的，命题得证。

（b）去掉 v_0 后 v_1 与 v_3 处在同一连通分支中，如图 6.5 所表示的那样，则 v_1 与 v_3 有一通路，其中点的颜色红黄交替出现，它与 v_0 构成一回路 C，也就是约当曲线，这时结点 v_2 处在曲线的内部而结点 v_5 则处在曲线的外部，v_2 与 v_5 的任何连线必与曲线 C 相交，与平面图的条件矛盾。因此约当曲线 C 必然将黑白集中的结点分成两个连通分支，使 v_2 和 v_5 分别处于两个连通分支中，于是问题回到（a），可将 v_2（或 v_5）所在的分支中的黑白色对换，于是与 v_0 邻接的 5 个结点也只着了 4 种颜色，v_0 就可着第 5 种颜色。 ▌

与定义 6.2 及定义 6.3 类似，我们也可得到图 G 边的正常着色、边 k 可着色以及边色数的定义。图 G 的边色数记作 $\Psi_e(G)$。

如图 6.6 结点上的标号表示结点着色种类号，边上标的数字则表示边着色的种类号，可见图（a）的点色数为 2 而边色数为 3，即 $\Psi_v(G_1)=2$，$\Psi_e(G_1)=3$，图（b）的点色数为 3 而边色数为 4，即 $\Psi_v(G_2)=3$，$\Psi_e(G_2)=4$。

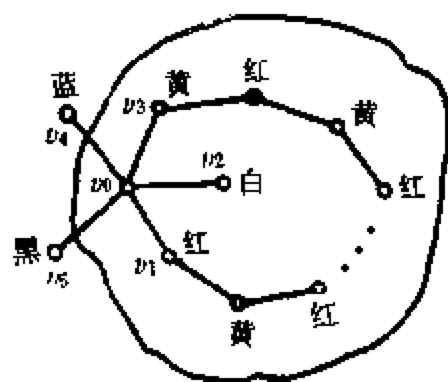


图 6.5

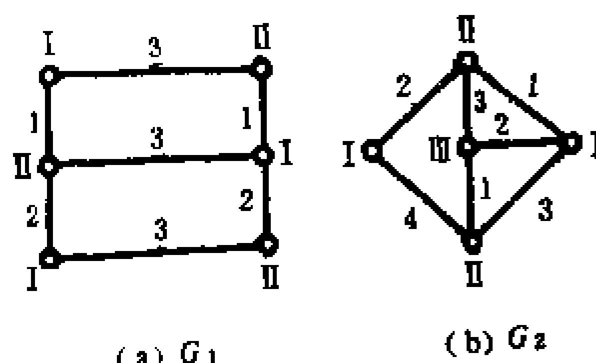


图 6.6

图 G 的边着色相当于对边集合的一种划分，每一种颜色的边集合构成 G 的一个匹配，一个图最多有多少边可以着同一颜色实际上是求图的最大基数匹配问题。

定理 6.3 无割边的简单连通 3 正则平面图都是边 3 可着色的。

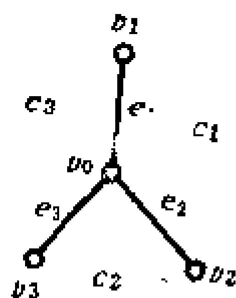


图 6.7

证：设 G 是所给的平面图，因为无割边所以每条边都分隔两个不同的面，如图 6.7 所示，根据四色定理，图的结点或面都是 4 可着色的，不妨设以 v_0 为公共点的三个面分别着上 C_1, C_2, C_3 三种颜色，现用边所分隔的两个面的颜色之和为该边着色，即用颜色（C_1+C_2）给边 e_2 着色，用（C_2+C_3）给 e_3 着色，用（C_1+C_3）给 e_1 着色，（显然用怎样的符号来表示颜色是无关紧要的），于是与每一个结点关联的 3 条边分别着上了三种颜色，图是边 3 可着色的。 ▌

§6.3 边的着色

前面我们讨论了平面图的着色，下面我们将进而讨论一般图的着色问题，也分为边的着色和点的着色两类，这一节先讨论边的着色。并首先给出两种特殊类型图边着色的特性。

在以后的讨论中，我们用 $\Delta(G)$ 表示图 G 的最大结点次数，并简记作 Δ。

定理 6.4 若 G 是偶图，则

$$\Psi_e(G)=\Delta \tag{6.1}$$

证：设 G 的两个互补结点子集为 V_1 和 V_2，若 $|V_1|<|V_2|$，则在 V_1 中增加一些结点成为 V_1' 使 $|V_1'|=|V_2|$，（如果 $|V_2|<|V_1|$ 则作相反处理使两子集的结点数相等），对 $x_i\in V_1'$ 及 $y_j\in V_2$，若 G 中无边（x_i, y_j），则增加一条边（x_i, y_j），通过以上的增添，图 $G=(V, E)$ 成为图 $G_\Delta=(V', E_1')$，显然 G_Δ 是 Δ 次正则偶图，由定理5.4的推论可知，它有一完美匹配 M_1，令 $E_2'=E_1'-M_1$，得到图 $G_{\Delta-1}=(V', E_2')$，则 $G_{\Delta-1}$ 是（$\Delta-1$）次正则偶图，它也有一完美匹配 M_2，如此继续下去可以得到 $M_1, M_2, \cdots M_\Delta$ 个完美匹配，每一个完美匹配可着一种颜色，便得到 G_Δ 的边 Δ 着色，即

$$\Psi_e(G_\Delta)=\Delta$$

由于 G 的结点最大次数为 Δ，所以在 G_Δ 的 Δ 个完美匹配中，每一个匹配至少含有 G 的一条边，从 Δ 个完美匹配中去掉增添的结点和边，得到图 G 的 Δ 个匹配，显然它也是边 Δ 着色的，即

$$\Psi_e(G)=\Delta$$ ▌

定理 6.5 对简单完全图 K_n，有

$$\Psi_e(K_n)=\begin{cases}n, & \text{当 } n \text{ 是奇数}\\ n-1, & \text{当 } n \text{ 是偶数}\end{cases} \tag{6.2}$$

证：设 $n=$偶数，显然满足定理 5.8 的条件，故图 G 存在完美匹配。记 G 的 n 个结点为 $v_0, v_1, \cdots, v_{n-1}$，取边集

$$M_1=\{(v_0, v_1), (v_2, v_{n-1}), (v_3, v_{n-2}), \cdots, (v_{\frac{n}{2}+1}, v_{\frac{n}{2}})\}$$

则 M_1 是 G 的一个完美匹配。同理按以上顺序作轮换（$v_1, v_2, \cdots, v_{n-1}$），将得到 $n-1$ 个边不重的完美匹配 $M_1, M_2, \cdots M_{n-1}$，且有

$$\sum_{i=1}^{n-1}|M_i|=\frac{1}{2}n(n-1)$$

对每一个完美匹配中的边着上一种颜色，得

$$\Psi_e(G)=n-1$$

设 n=奇数，这时在图 G 上增加一个点 v_n 并在 v_n 与其余 n 个结点之间都连接一条边，图 G 成为 G'，因此 G' 是结点数为 $n'=n+1$ 的完全图，而 n' 为偶数，它有 $n'-1=n$ 个完美匹配，即

$$\Psi_e(G')=n$$

现将 G' 中结点 v_n 及与 v_n 关联的边去掉，图 G' 恢复成 G，而在 n 个完美匹配中，每一个匹配只去掉一条边 (v_n, v_i)，$i=0, 1, \cdots n-1$，结果仍然是 n 个边不重的匹配，它们的并集即是 G 的边集，因此

$$\Psi_e(G)=n \qquad \blacksquare$$

以上给出的是求两个特殊图边色数的公式，至于一般图的边色数是多少呢？首先我们看到，如果图的最大结点次数为 Δ，意味着至少存在一个结点它有 Δ 条关联边，这 Δ 条边必须着不同颜色，因此图的边色数不可能小于 Δ，即

$$\Delta\leqslant\Psi_e(G)$$

其次，任何一个简单图的边色数不会超过结点数相同的完全图的边色数，由式（6.2），则一般图的边色数应有

$$\Psi_e(G)\leqslant\Delta+1=n$$

基于以上考虑，人们自然会得出如下关系，即

$$\Delta\leqslant\Psi_e(G)\leqslant\Delta+1$$

1964 年 Vizing 证明了这一结论。

定理 6.6 （Vizing）对任意简单图 G，

$$\Delta\leqslant\Psi_e(G)\leqslant\Delta+1 \tag{6.3}$$

证：$\Delta\leqslant\Psi_e(G)$ 是显然的，下面只证 $\Psi_e(G)\leqslant\Delta+1$。对边数用归纳法证明如下。

设 $|E|=m$，当 $m=1$ 时定理显然成立。设 $|E|=m-1$ 时定理成立，证明 $|E|=m$ 时命题亦成立。此时图为 G。

设在 G 中只有边 (u, v) 尚未着色，其余 $m-1$ 条边已正常着色，并且色数不超过 $\Delta+1$。

我们约定：如与结点 v_i 关联的边已着颜色 C，则称色 C 在结点 v_i 上出现，或称结点 v_i 有色 C，如与 v_i 关联的所有边都没有着颜色 C，则称色 C 没有在结点 v_i 上出现，或称结点 v_i 缺色 C。

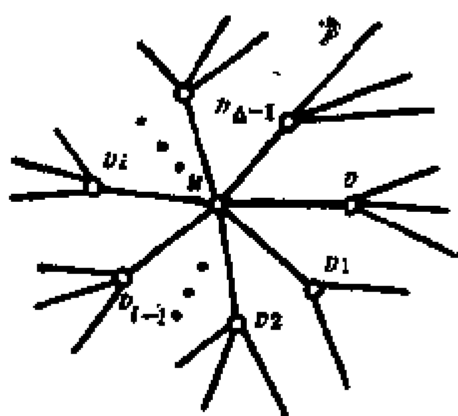


图 6.8

现在考查结点 u，由于 G 的结点最大次数为 Δ，故与 u 邻接的结点最多有 Δ 个，除与 v 邻接外，不妨设邻接点还有 $v_1, v_2, \cdots v_{\Delta-1}$，而且 u 与这 $\Delta-1$ 个结点已正常着色，即 u 与这些结点的连接边都已着上了不同的颜色，如图 6.8 所示，我们进而考查与 u 邻接的 Δ 个结点，可以得出：

（1）与这些结点关联的边不超过 Δ 条，给这些边着不同颜色不会超过 Δ 种，而题设色数可以达到 $\Delta+1$，因此每一结点至少有一种颜色不在其上出现，即每一结点至少缺一种颜色。

（2）当这些结点与u连接的边未着色时，则其中任意两个结点都不可能只缺相同的一种颜色。因为如果v_i只缺色C，则边(u, v_i)必须着C色，若v_y也只缺色C，则边(u, v_y)也必须着C色，于是这两条都与结点u关联的边都着了相同的颜色C，与正常着色矛盾。

基于以上分析，我们先考查结点v，设v点缺C_1色，若u点也缺C_1色，则可将未着色的边(u, v)着C_1色，使得色数没有超过$\Delta+1$，命题得证。

如果u并不缺C_1色，说明必然有与u关联的一条边着了C_1色，不妨设是边(u,v_1)，继而考查结点v_1，因为v_1也一定缺一种颜色，设为C_2，如果u也缺C_2，则可将边(u, v_1)改着C_2色而将色C_1着边(u, v)，于是边(u, v)得到正常着色，命题亦得证。若u并不缺C_2色，则必然有一条与u关联的边着了色C_2，设是边(u, v_2)，可用以上的分析方法考查结点v_2，如此v_2，v_3，…逐个考查下去，只要考查到与u邻接的结点v_i缺的颜色也是u缺的颜色，则可将边(u,v_i)着上这种颜色，而将原来给边(u,v_i)着的色C_i退回去给边(u, v_{i-1})着色，如此逐个退回重新着色，最后也可使边(u, v)着上色C_1而达到正常着色。由于u的次数是有限的，考查到与u邻接的最后一个结点$v_{\Delta-1}$，则$v_{\Delta-1}$缺的颜色必然也是u缺的颜色，可将这种颜色给边$(u, v_{\Delta-1})$着色，并按上述步骤依次退回重新着色而达到使边(u,v)正常着色的目的。 ▮

推论：对无环多重图G，有

$$\Delta \leqslant \Psi_e(G) \leqslant \Delta + M \tag{6.4}$$

其中M为G中连接两个结点之间的最大边数（称为G的最大重边数）

由定理6.6知，任意一个简单图G，它的边色数只有两种可能值，即$\Psi_e(G)=\Delta$或$\Psi_e(G)=\Delta+1$，如果是前者，称图是第一型的而称后者的图是第二型的。一个图究竟属第一型还是第二型，除了一些特殊图之外（如上面讲的偶图或完全图），尚无一个普遍有效的判定法则。仍是一个尚待解决的课题。

定理 6.7 有n个结点m条边的简单图G，若

$$m > \Delta \cdot \left\lfloor \frac{n}{2} \right\rfloor \tag{6.5}$$

则G是第二型的。

证：用反证法，设G是第一型的，则用Δ种颜色即可使图的边正常着色，着同一种颜色的边即是图的一个匹配，由于图有n个结点，显然最大基数匹配边数也不会大于$\left\lfloor \frac{n}{2} \right\rfloor$，故$\Delta$个匹配总的边数不会超过$\Delta \cdot \left\lfloor \frac{n}{2} \right\rfloor$，即

$$m \leqslant \Delta \cdot \left\lfloor \frac{n}{2} \right\rfloor$$

和假设矛盾，故G不是第一型的。 ▮

但应注意，这一定理给出的条件是充分而非必要条件。例如图6.9所示的彼特森图是第二型的，并未满足这一条件。

研究图的边着色问题，有它的实际意义，所谓排课表问题，就是一个典型的应用例子。

例如某所学校有n名教师$x_1, x_2, \cdots, x_n$和m个班级$y_1, y_2, \cdots, y_m$，已确定教师x_i每

周要给班级 y_j 上 P_{ij} 节课，在教室足够的前提下，问应怎样制订课时表使上课的节数尽可能少。

设 $X=\{x_1, x_2, \cdots, x_n\}$，$Y=\{y_1, y_2, \cdots, y_m\}$，令 X 和 Y 为偶图的两个互补结点子集，若 x_i 给 y_j 上P_{ij}课，则在结点 x_i 与 y_j 之间连接 P 条平行边，如图 6.10。显然，同

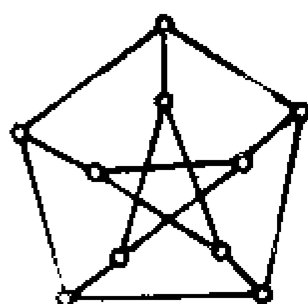


图 6.9

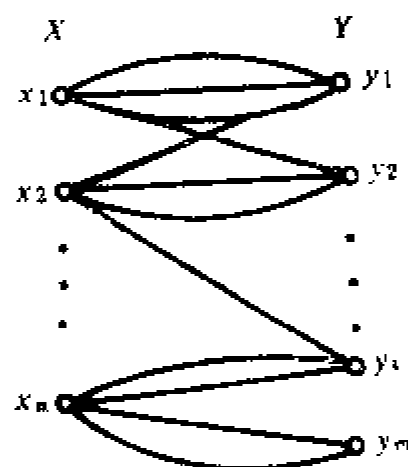


图 6.10

一节课里，一位教师只能教一个班级，而每个班级最多只能由一位教师上课，所以一节课里教师和班级构成一个匹配，排课表问题就是把偶图划分为匹配并使匹配的个数尽可能地少，相当于寻求边着色的色数，已知偶图的边色数是 Δ，因此，如果没有教师教多于 P 节的课以及没有班级上多于 P 节的课，那么就可制订 P 课时的课表，也是最少节数的课时表。

以上是在教室足够的条件下，课时表的安排是比较简单的，如果教室比较紧张，如何在有限教室的条件下制订一个较为完善的课时表呢？

例如，有四位教师和五个班级，教师 x_i 给班级 y_j 开 P_{ij} 节课可用讲课矩阵 $P=(p_{ij})$来表示如右。

$$P=\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array}\begin{array}{c} \begin{array}{ccccc} y_1 & y_2 & y_3 & y_4 & y_5 \end{array} \\ \left[\begin{array}{ccccc} 2 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right] \end{array}$$

可以有两种排课方案均能满足开课的要求，如表 6.1 的(a)和(b)，在(a)表中，第一节有四个班级上课，因此

表 6.1

(a)

教师 \ 班级 \ 节次	1	2	3	4
x_1	y_1	y_1	y_3	y_4
x_2	y_2	\	y_4	\
x_3	y_3	y_4	\	y_2
x_4	y_4	y_5	\	\

(b)

教师 \ 班级 \ 节次	1	2	3	4
x_1	y_4	y_1	y_3	y_1
x_2	y_2		y_4	\
x_3	y_3	y_4	\	y_2
x_4	\	y_5	\	y_4

需要四个教室，而在(b)表中，每一节课都只有三个班级在上课，故只需三个教室。由此说明在课时节数一定的前提下，如果安排恰当，可以少用教室。

另一方面，如果只有两个教室可供使用，显然上面两种课表都不能满足开课要求，必须增加课时节数，那么应该安排多少节课才恰当呢？下面的定理作了回答，现在先介绍一个引理。

引理 6.1 设M和N是图G的两个边不重的匹配，并且$|M|>|N|$，则G中存在两个边不重的匹配M'和N'，使得$|M'|=|M|-1$，$|N'|=|N|+1$，且$M'\cup N'=M\cup N$。

证：由定理5.2的证明可知，$M\oplus N$的每个连通分支，或者是由M和N的边交替出现构成的回路，或者是M和N的边交替出现的路径，由于$|M|>|N|$，必有一条交替路$P=v_0, v_1, v_2, \cdots, v_{2n-1}, v_{2n}, v_{2n+1}$，其路径长度为奇数且两个端点$v_0$和$v_{2n+1}$是$M$饱和点而非$N$饱和点，即$P$中属于$M$的边比属于$N$的边多一条，由于$M$与$N$边不重(即无公共边)，$M\oplus N=M\cup N$，现在我们把$P$中边的隶属关系改变，即将原属于$N$的边改为属于$M$，原属于$M$的边改为属于$N$，于是$M$变为$M'$，$N$变为$N'$，即

$$M'=[M-\{(v_0, v_1), (v_2, v_3)\cdots, (v_{2n}, v_{2n+1})\}]\cup\{(v_1,v_2), (v_3, v_4), \cdots, (v_{2n-1}, v_{2n})\}$$

$$N'=[N-\{(v_1, v_2), (v_3, v_4), \cdots, (v_{2n-1}, v_{2n})\}]\cup\{(v_0, v_1), (v_2, v_3), \cdots, (v_{2n}, v_{2n+1})\}$$

显然有

$$M'\cap N'=\phi,\ |M'|=|M|-1,\ |N'|=|N|+1$$ ▮

定理 6.8 对偶图G，若$P\geqslant\Delta$，则存在G的P个边不重的匹配$M_1, M_2, \cdots, M_P$，使得

$$E\doteq M_1\cup M_2\cup\cdots\cup M_P \tag{6.6}$$

并且对于$1\leqslant i\leqslant P$，有

$$\left\lfloor\frac{|E|}{P}\right\rfloor\leqslant|M_i|\leqslant\left\lceil\frac{|E|}{P}\right\rceil \tag{6.7}$$

证：由定理6.4知，偶图G可以划分为Δ个边不重的匹配$M_1', M_2', \cdots, M_\Delta'$，对$P\geqslant\Delta$，可以将这$\Delta$个匹配再细分为$P$个边不重的匹配$M_1, M_2, \cdots, M_P$，使得

$$E=M_1\cup M_2\cup\cdots\cup M_P$$

式(6.6)表明，在G的P个匹配中的任意两个匹配M_i和M_j，它们的边数最多相差1，这是可以实现的，因为对边数相差超过1的任意两个匹配M_i和M_j，可以反复应用引理6.1使最后得到G的P个匹配满足式(6.6)和(6.7)。 ▮

式(6.7)为我们恰当地制订课表提供了理论依据，式中$|E|$相当于每天上课总时数，P为每天安排的上课节数，$|M|$为每节课上课需要的教室数，如上面的例子，当总课时为11节课时，如果每天安排四节课，需要教室三间，如果只有两间教室可供使用，则必须至少安排六节课才能满足要求。

§6.4 独立集、支配集、覆盖和团

为了给下一节图的结点着色作准备，这一节先介绍独立集和支配集的概念。

定义 6.4 对图$G=(V, E)$的结点子集$S\subseteq V$，如果

(1) S中的任意两个结点均不相邻，则称S为G的一个独立集。

（2）若S是G的独立集，且任一结点$v \in V-S$必与S中至少一个结点相邻，则称S为G的极大独立集。

（3）若S是G的独立集，且不存在G的另一独立集S'满足$|S'|>|S|$，则称S为G的最大独立集。

（4）G的最大独立集S的基数$|S|$，称为G的独立数，记作$I(G)$。

如图6.11的图G，$\{a\}$，$\{a, c\}$，$\{a, c, d\}$，$\{a, c, f\}$等都是独立集而$\{a, e\}$，$\{b, f\}$等是极大独立集，$\{a, c, d, f\}$，$\{a, c, d, g\}$则是最大独立集，这里$I(G)=4$。

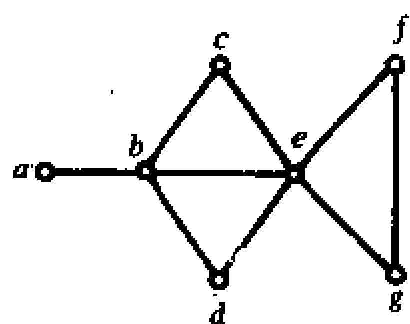


图 6.11

由独立集的定义可知：

（1）图G的每一个结点构成一个独立集。

（2）极大独立集不是唯一的，它的基数不一定是最大的，但它的元素数目已达到极限，即不可能再加入其他结点而不破坏它的独立性。

（3）最大独立集必然也是极大独立集而且元素数目是最多的。

（4）任一完全图K_n的独立数$I(K_n)=1$

（5）偶图G只有两个极大独立集，即是它的两个互补结点子集V_1和V_2。

图的独立集与图的匹配相类似，都是从独立性的概念来定义的，前者是图的不相邻的结点集合，后者则是图的不相邻的边集合，虽然如此，独立集却没有类似匹配理论的独立集理论。而且，目前还没有一个求图的最大独立集的有效算法。从理论上讲要求图的最大独立集并不困难，只要采取逐点试探法，最终总能找到图的极大独立集，所谓逐点试探法就是从任一结点开始，然后每次增加一个结点，使所加的结点与已有的结点均不相邻，如此反复进行直到最后一个结点为止，便得到图的极大独立集，比较所有的极大独立集，基数最大的就是最大独立集。但是采用这种算法，其计算复杂性是指数型的而不是多项式级的。

下面介绍一种应用布尔变量的运算法则求最大独立集的算法。为此先介绍布尔表达式及布尔运算的概念。

称参加布尔运算的变量为布尔变量，我们这里需要用到的布尔运算有两种，即布尔乘法与布尔加法。

布尔变量a与b的布尔乘也叫逻辑“与”，记作$a \cdot b$，（以后简写作ab）。

布尔变量a与b的布尔和也叫逻辑“或”，记作$a+b$。

布尔运算满足以下基本规律：

1．幂等律　$a \cdot a=a$，$a+a=a$

2．结合律　$a \cdot (b \cdot c)=(a \cdot b) \cdot c$

$a+(b+c)=(a+b)+c$

3．交换律　$a \cdot b=b \cdot a$，$a+b=b+a$

4．分配律　$a \cdot (b+c)=a \cdot b+a \cdot c$

$a+(b \cdot c)=(a+b) \cdot (a+c)$

5．吸收律　$a \cdot (a+b)=a$

$a+(a \cdot b)=a$

采用布尔运算的代数表达式称为布尔表达式。有两种基本型式，其中一种称为和之积式，即

$$A=\prod_{i=1}^{n}(a_i+b_i+\cdots+t_i)$$
$$=(a_1+b_1+\cdots+t_1)(a_2+b_2+\cdots+t_2)\cdots(a_n+b_n+\cdots+t_n)$$

例 6.1 化简下列布尔表达式

$$A=(b+d)(b+d+c)+(a+b+d)(a+c+d)$$

解：因 $(b+d)(b+d+c)=b+d$ 吸收律

而 $(a+b+d)(a+c+d)=a(a+c+d)+b(a+c+d)+d(a+c+d)$ 分配律

$=a+b(a+c+d)+d$ 吸收律

$=a+ba+bc+bd+d$ 分配律

$=a+bc+d$ 吸收律

故 $A=b+d+a+bc+d$

$=a+b+d$ 交换律，幂等律，吸收律

下面介绍求图的极大独立集的算法。

设 I 是图 $G=(V,E)$ 的一个极大独立集，它的补集 $\bar{I}=V-I$，显然，对图的任意一条边 $(u,v)\in E$，$\bar{I}$ 一定含有它的一个端点或者两个端点，所以如果能求出对每一条边都满足这一条件的最小集合 $\bar{I}$ 也就可以求得 $I(G)$，为此我们构造一个和之积的布尔表达式

$$B=\prod_{(u,v)\in E}(u+v) \tag{6.8}$$

其中每一项都是 $\bar{I}$ 的最小集合并且包含 E 的每一条边的至少一个端点，取这些最小集的补即得极大独立集，从中很容易求出 $I(G)$。现举一例说明如下

例 6.2 试用布尔算法求图 6.11 的极大独立集及 $I(G)$

解：
$$B=\prod_{(u,v)\in E}(u+v)$$
$$=(a+b)(b+c)(b+e)(b+d)(c+e)(d+e)(e+f)(e+g)\times(f+g)$$
$$=bef+beg+begf+bcdef+bcdeg+bcdgf+acdef+$$
$$+acdeg+acdegf$$
$$=bef+beg+bcdfg+acdef+acdeg$$

取上式各项结点的补集即得极大独立集如下

$$V-\{b,e,f\}=\{a,c,d,g\}$$
$$V-\{b,e,g\}=\{a,c,d,f\}$$
$$V-\{b,c,d,f,g\}=\{a,e\}$$
$$V-\{a,c,d,e,f\}=\{b,g\}$$
$$V-\{a,c,d,e,g\}=\{b,f\}$$

可见 $I(G)=4$

定义 6.5 对图 $G=(V,E)$ 的结点子集 $S\subseteq V$，如果

（1）V 中任一结点或属于 S，或至少与 S 的一个结点邻接，则称 S 为 G 的一个支配集。

（2）若S是G的支配集，且S的任一真子集都不是G的支配集，则称S为G的极小支配集。

（3）在所有极小支配集S中取$|S|$的最小值，称为G的支配数，记作$D(G)$。

如图 6.12 的图G，$\{a, b, d, e\}$，$\{a, c, d, f\}$等都是支配集，而$\{b, e\}$，$\{d, f\}$，$\{a, c, f\}$等则是极小支配集，这里$D(G)=2$

由支配集的定义可知：

（1）图的结点集合本身就是图的一个支配集，它是最大支配集

（2）完全图的每一个结点构成图的一个支配集。

（3）极小支配集不是唯一的，它的结点数不一定是最少的，但它的元素数目已达到极小限度，即如果去掉其中任一结点就不再成为支配集。

（4）任一支配集必然包含一极小支配集。

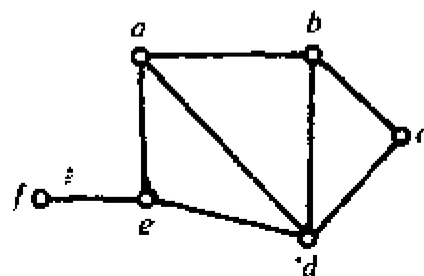


图 6.12

支配集也称为图的点支配，因为图的所有结点一定与支配集中一个或多个结点邻接，从这个意义上说，支配集支配了整个图的点，而极小支配集则是通过极少数的点支配整个图的点，因而更具有实际意义。例如，若图表示的是一张作战地图，各结点表示城市，边表示它们之间的通路，那么只要在极小支配集所包含的结点上配置足够的火力，就可控制和封锁所有城市，显然，它的战略意义是极为重要的。

和独立集一样，目前也没有一个求图的极小支配集的有效算法。我们仍然采用布尔运算的法则求图的支配数。

令$Adj(v_i)$表示与结点v_i邻接的结点集合，φ_i表示结点v_i以及与v_i邻接的所有结点的布尔和，即

$$\varphi_i=v_i+\sum_{v_j\in Adj(v_i)} v_j \tag{6.9}$$

构造布尔表达式

$$A=\prod_{i=1}^{n}\varphi_i \tag{6.10}$$

则A中每一项即为图的一个极小支配集。

例 6.3 试用布尔算法求图 6.12 的极小支配集及$D(G)$

解：由图可得

$$\varphi_a=a+b+d+e$$
$$\varphi_b=a+b+c+d$$
$$\varphi_c=b+c+d$$
$$\varphi_d=a+b+c+d+e$$
$$\varphi_e=a+d+e+f$$
$$\varphi_f=e+f$$

故 $A=(a+b+d+e)(a+b+c+d)(b+c+d)(a+b+c+d+e)(a+d+e+f)(e+f)$

$=(a+b+d+e)(b+c+d)(e+f)$ 　　吸收律

$=be+de+ec+fb+fd+fac+ace$

以上 7 项对应图的 7 个极小支配集：$\{b, e\}$，$\{d, e\}$，$\{e, c\}$，…，$\{a, c, e\}$，其中基数最小的是 2，故 $D(G)=2$。

一般 φ_i 中至少含有两项，所以对布尔表达式 A 的运算至少超过 2^n 次，因此，当图的结点数 n 很大时，即便使用高速计算机来计算，也是难以完成的。

独立集和支配集都是图的结点子集，它们之间有没有内在的联系呢？下面的定理，描述了它们之间的关系。

定理 6.9 图的独立集 S 也是图的支配集的充要条件是 S 是极大独立集。并有

$$D(G)\leqslant I(G) \tag{6.11}$$

证：若 S 是极大独立集，则任一结点 $v\in V-S$，必与 S 中至少一个结点相邻，由支配集的定义知，S 是图的支配集。

反之，若独立集 S 也是支配集，则 S 外的任一结点必与 S 中至少一个结点相邻，由极大独立集的定义知，S 必是极大独立集。并且由独立数与支配数的定义显然可得

$$D(G)\leqslant I(G)$$ ▌

除了支配集之外，人们还提出了点覆盖的概念，定义如下。

定义 6.6 对图 $G=(V, E)$ 的结点子集 $K\subseteq V$，如果

（1） G 的任意一条边都至少有一个端点属于 K，则称 K 为 G 的一个点覆盖。

（2） K 是 G 的点覆盖，若对任一结点 $v\in K$，$K-\{v\}$ 不再是点覆盖，则称 K 为 G 的极小点覆盖。

（3） K 是 G 的点覆盖，且 G 不存在另一点覆盖 K' 满足 $|K'|<|K|$，则称 K 为 G 的最小点覆盖。

（4） G 的最小点覆盖的基数，称为 G 的覆盖数，记作 $C(G)$。.

G 的点覆盖常简称为 G 的覆盖。

如图 6.13，$\{a, b, d, e, g\}$ 是图的一个覆盖，$\{b, e, c, f\}$ 是图的一个极小覆盖，而 $\{a, d, g\}$ 则是图的最小覆盖。

定理 6.10 设图 $G=(V, E)$，$S\subseteq V$，则 S 是 G 的独立集当且仅当 $(V-S)$ 是 G 的覆盖。

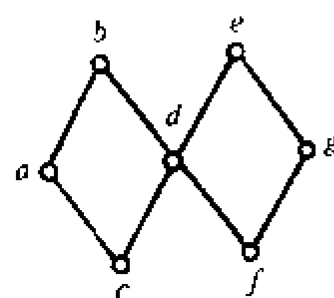


图 6.13

证：设 S 是 G 的独立集，则 S 中任意两结点均不相邻，即 G 的任一条边最多有一个端点在 S 中，因而至少有一个端点在 $(V-S)$ 中，所以 $(V-S)$ 是 G 的覆盖。

反之，若 $(V-S)$ 是 G 的覆盖，则 G 的任一条边至少有一个端点在 $(V-S)$ 中，故最多有一个端点在 S 中，即 S 中的结点在不邻接，所以 S 是 G 的独立集。 ▌

推论：
$$I(G)+C(G)=|V| \tag{6.12}$$

证：设 S 是 G 的最大独立集，K 是 G 的最小覆盖，则 $(V-S)$ 是 G 的覆盖而 $(V-K)$ 是 G 的独立集，因此

$$|V-S|\geqslant C(G), \qquad |V-K|\leqslant I(G)$$

而
$$|V-S|=|V|-I(G), \ |V-K|=|V|-C(G)$$

即
$$|V|\leqslant I(G)+C(G)\leqslant |V|$$

故
$$I(G)+C(G)=|V|$$ ▌

下面介绍与独立集和支配集有关的一个概念——团，由于含自环和平行边的多重图均可化为简单图进行讨论而不影响问题的性质，下面只讨论简单图。

定义 6.7 若简单图$G=(V, E)$的子图S是完全图，则称S是G的团。

可以将图的结点划分为若干个子集，使每一子集都构成团。将图划分为最少个数的团，此时团的个数记作$B(G)$。

如图6.14的简单图，可以划分为4个团，即

$\{a, b\}$，$\{c, d\}$，$\{e\}$，$\{f\}$也可划分为3个团，即

$$\{a, e, f\}, \{b, c\}, \{d\}$$

而且它是团的最少划分，即$B(G)=3$。

显然团的最多数目就是图的结点数，此时图的每一结点构成一个团。

由定义显然可知，S是G的团当且仅当S是G的补图$\bar{G}$的独立集。因此这两个概念是互补的。

定理 6.11 对任意图G有$I(G)\leqslant B(G)$。而且若I是一个独立集，P是图的一个团划分，且$|I|=|P|$，则$|I|=I(G)$和$|P|=B(G)$。

证：由定义可知，独立集的结点在任何一个团中不会多于一个，因此

$$|I|\leqslant|P|$$

上式对任意的独立集和任意的团划分均成立，故

$$I(G)=\max|I|\leqslant\min|P|=B(G)$$

如果$|I|=|P|$，显然有$|I|=I(G)$及$|P|=B(G)$。 ∎

由于团的内部结点是彼此邻接的，一个团结点数的多或少，对图的独立数$I(G)$和支配数$D(G)$的大小必然有直接的影响。从直观上可知，要使图没有某一结点数目的团，图的边数一定有一个极限。下面介绍由Turan（1941）提出的一个著名的定理，它给出了有n个结点而不包含完全图K_{l+1}的团的简单图所能具有的边数的上界。在引出这一定理之前，先介绍一些预备知识。

（1）结点的次序列

将图的结点按其次数递增的顺序排列，如图6.15的图，结点的次序列为(2,2,3,3,4)。

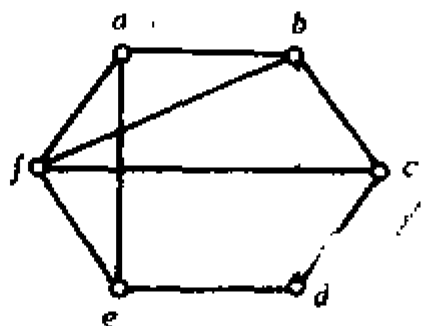


图 6.14

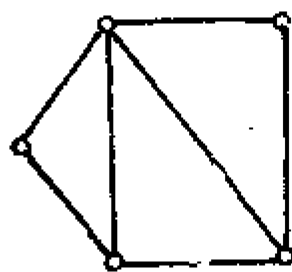


图 6.15

（2）G_1次弱于G_2

若图G_1和G_2的结点一一对应，且G_1的结点的次数均小于等于G_2中与之对应的结点的次数，则称图G_1次弱于G_2。

例如偶图$K_{2,3}$的结点次序列为（2,2,3,3,3），可见它次弱于图6.15所示的图。

（3）K分图和完全K分图

它是偶图（二分图）概念的推广，如果图G的结点可以划分为K个非空子集，使图的

任何一条边的两个端点均不同在一个子集中，则称图为K分图。如果K分图的每个结点均和不在同一子集的所有结点邻接，则称之为完全K分图。

如图6.16的（a）为3分图，而（b）则是完全3分图。

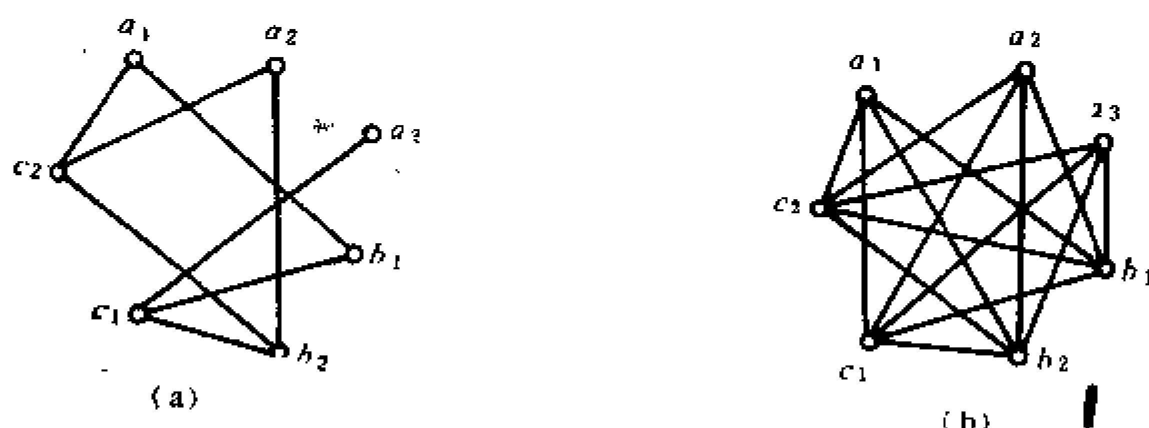


图 6.16

定理 6.12 若简单图G不包含K_{i+1}的团，则G次弱于某一完全i分图P。并且，若G和P有相同的次序列，则G与P同构，即$G\cong P$。

证：对i用归纳法，当$i=1$时，则G不含任何边，定理显然成立。假设定理对所有的$i<j$成立，并设G是不包含完全子图K_{j+1}的简单图，设结点u是G中次数最大的结点，G_1是所有与u邻接的结点构成的子图，由于G不包含K_{j+1}，所以G_1不包含K_j，因此由归纳假设，G_1必然次弱于某一完全$(j-1)$分图P_1，记G_1的结点集为V_1及$V_2=V-V_1$，V是G的结点集，设G_2是由V_2组成但不含任何边的图（即零图），现在考察G_1与G_2的联图（即在G_1的每一结点与G_2的所有结点之间连一条边所构成的图），并记作$J(G_1,G_2)$，显然，在$J(G_1, G_2)$的结点子集V_2中的任一结点，它的次数都等于图G结点的最大次数，而在V_1中的结点的次数至少与它原来在G中的次数相同，即G和$J(G_1, G_2)$的结点一一对应，且G的结点次数均小于等于$J(G_1, G_2)$中与之对应的结点的次数，所以G次弱于$J(G_1, G_2)$。因为G_1次弱于完全$(j-1)$分图P_1，所以G必次弱于完全j分图$P=J(P_1, G_2)$，定理的第一部分得到证明。

现在假设G和P有相同的次序列，则G_1和P_1有相同的次序列，由归纳假设G_1与P_1同构，而且G也必然与$J(G_1, G_2)$有相同的次序列，于是在G中，V_1中的每一结点必与V中任一结点连接，因此G与P同构。 ▌

图6.17从直观上描述了上述定理的证明过程。

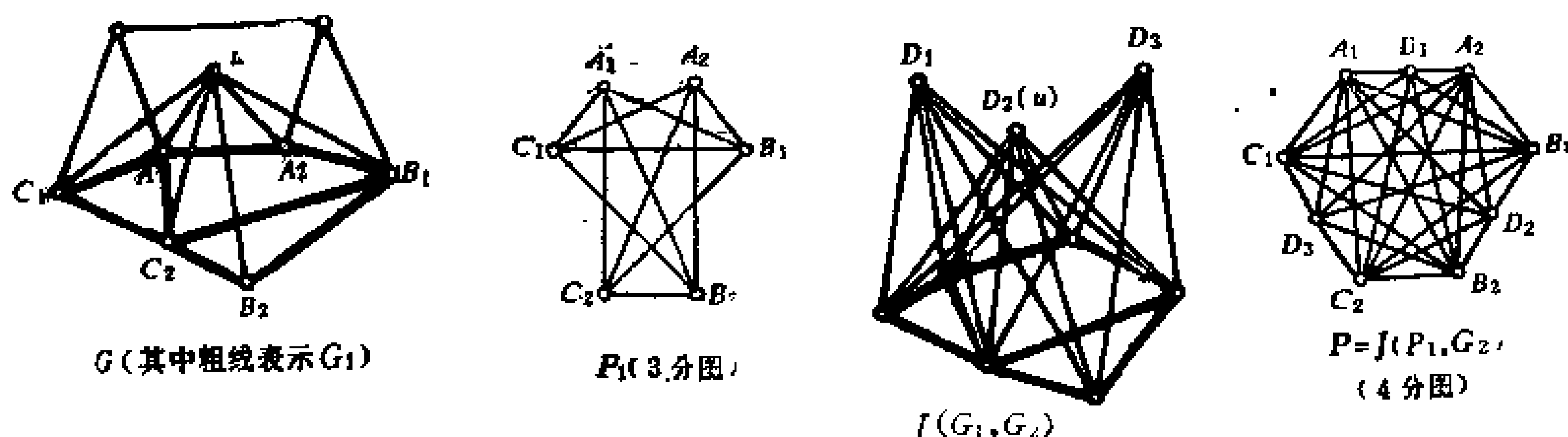


图 6.17

设$T_{j,n}$表示有n个结点的完全j分图，并且它的各部分结点的数目尽可能相同，如图6.17中的P即是图$T_{4,9}$。下面用$E(G)$表示图G的边集。

定理 6.13 (Turan) 若G是简单图，并且不包含K_{j+1}，则$|E(G)|\leqslant|E(T_{j,n})|$，此外，仅当$G$与$T_{j,n}$同构时有$|E(G)|=|E(T_{j,n})|$。

证：若G是不包含K_{j+1}的简单图，由定理6.12知G次弱于某一完全j分图P，因此

$$|E(G)|\leqslant|E(P)|$$

而且可以证明（留作习题）

$$|E(P)|\leqslant|E(T_{j,n})|$$

因此
$$|E(G)|\leqslant|E(T_{j,n})| \tag{6.13}$$

现在（6.13）式中的等号成立，则必然有$|E(G)|=|E(P)|$，由定理6.12知G与P同构，而且有$|E(P)|=|E(T_{j,n})|$，则P与$T_{j,n}$也同构，因此G与$T_{j,n}$同构。 ▮

§6.5 点的着色

对一般图的结点正常着色，也有重要的实际意义。例如一家化工厂生产中需要n种原料c_1, $c_2\cdots$, c_n，有些原料不能相互接触，否则将引起火灾或爆炸，为此，工厂必须建立若干彼此隔离的仓库以便分别贮存相互不能接触的原料。试问：这样的仓库至少要建立多少座方能满足要求？

如果我们构造一个图$G=(V,E)$，令它的每一节点代表一种原料，即$V=\{c_1,c_2,\cdots,c_n\}$，若原料c_i和c_j不能接触，则在结点c_i和c_j之间连接一条边，可以看出，求仓库的最小数相当于求图的点色数。

又如学校在学期结束时对考试的安排，设有n门考试课程，学生必须参加他选修的所有课程的考试，但每一场学生只能参加一门课程的考试，问应如何安排方能使考试的场次最少？如果用结点v_1, v_2, $\cdots$, v_n表示考试的课程，若课程v_i和v_j有学生同时选修，则在两结点之间连接一条边，可见，求考试的最少场次也化归为求图的点色数问题。

§6.3 图的边着色问题也可转化为相应的点着色问题。例如，对图6.18（a）的边着色，可以转化为相应图的点着色，方法是

（1）在G的每一条边e_i上置一结点v_i（v_i不得是e_i的端点）

（2）若边e_i和e_j邻接，则在它们的对应结点v_i和v_j之间连接一条边。

（3）由对应结点及连接边组成的图G'即是G的对应图，对G的边着色相当于对G'的点着色。

图6.18（b）即为按以上步骤得到的图G'。

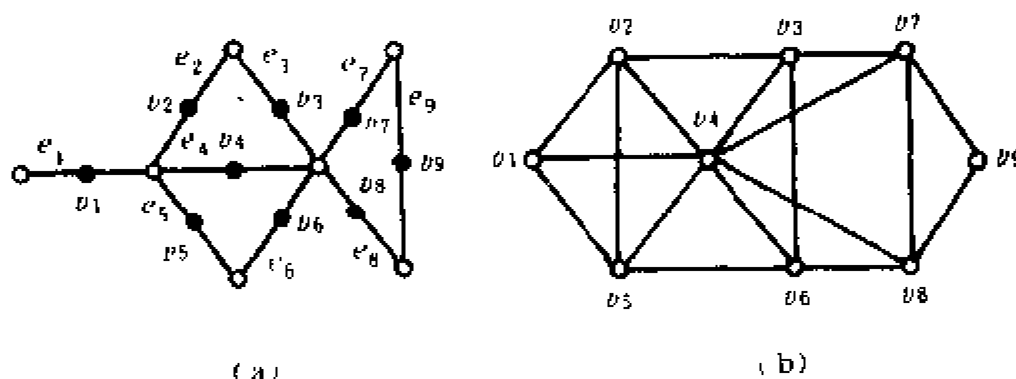


图 6.18

现在先讨论一些特殊图的点色数，由图的特征及色数的定义，显然可得

（1）如果图G是零图，则$\psi_v(G)=1$

（2）对完全图K_n，$\psi_v(G)=n$

（3）若图G是有n个结点的简单回路，则

$$\psi_v(G)=\begin{cases}2，当 n 为偶数\\3，当 n 为奇数\end{cases}$$

（4）对偶图$K_{n,m}$（包括树），$\psi_v(K_{n,m})=2$

（5）对k分图G，$\psi_v(G)=K$

由此可知，对图的点着色，也就是求图的最小点划分，即将图的结点划分为若干独立集，使独立集的数目最少，因为在一个独立集内的结点是彼此不相邻的，所以独立集内的所有结点可以着同一颜色。

定理 6.14 若图$G=(V,E)$有$|E|>1$，则点色数$\psi_v(G)=2$的充要条件是G没有奇次回路。

证：若G不含奇次回路，由定理5.1可知G是偶图，故$\psi_v(G)=2$。

反之，若$\psi_v(G)=2$，证明G没有奇次回路，用反证法，设G有奇次回路，由上面的（3）可知$\psi_v(G)\geqslant 3$，与题设矛盾。 ▌

定理 6.15 设G是一个有n个结点的简单图，则

$$I(G)\geqslant\frac{n}{\psi_v(G)} \tag{6.14}$$

证：设$\psi_v(G)=K$，即图至少需要着K种颜色，设为c_1，$c_2\cdots$，c_k，着同一种颜色C_i的结点集合S_i是G的一个独立集，显然有

$$|S_i|\leqslant I(G)$$

故
$$n=\sum_{i=1}^{K}|S_i|\leqslant K\cdot I(G)$$

即
$$\frac{n}{\psi_v(G)}\leqslant I(G)$$ ▌

定理 6.16 对任意图G，有

$$\psi_v(G)\leqslant\Delta+1 \tag{6.15}$$

证：对图的结点数n作归纳证明，当$n=1$，即图只有一个结点，$\psi_v(G)=1$，定理显然成立。设结点数为$n-1$时定理成立，现增加一个结点v_0，则与v_0邻接的结点最多有Δ个，即使这些结点都着不同颜色，也不会超过Δ种颜色，则给v_0着不同颜色，色数不会超过$\Delta+1$种，定理亦成立。 ▌

将定理6.16与定理6.6（Vizing）比较，两者都是企图为图的色数给定一个范围(一个是点色数，一个是边色数)，然而定理6.16给出的是色数的上界，有时比实际值要大得多。例如一个平面图，无论结点的次数有多大，点色数不会超过4，而偶图的点色数只是2，因此，这一定理的效果是相当弱的，相对来说，Vizing定理的效果要强得多。下面不加证明地引入由Brooks提出的一个定理，对色数的上界略有改进。

定理 6.17 若简单连通图G不是完全图，且$\Delta\geqslant 3$，则

$$\psi_v(G)\leqslant\Delta \tag{6.16}$$

下面介绍求图的点色数的几种方法。

一、凝缩、并接法

设图$G=(V, E)$，v_i与v_j是G的两个不相邻的结点，若将v_i与v_j凝缩为一点z，图G变成G'，称为图的凝缩。凝缩时原与v_i或v_j关联的边成为与z关联，平行边只须保留一条。若在v_i与v_j之间添上一条边，图G变成G''，称为图的并接。

如图6.19（a）的图G，v_2与v_4是两个不相邻结点，通过凝缩得到（b）的图G'，通过并接则得到（c）的图G''。

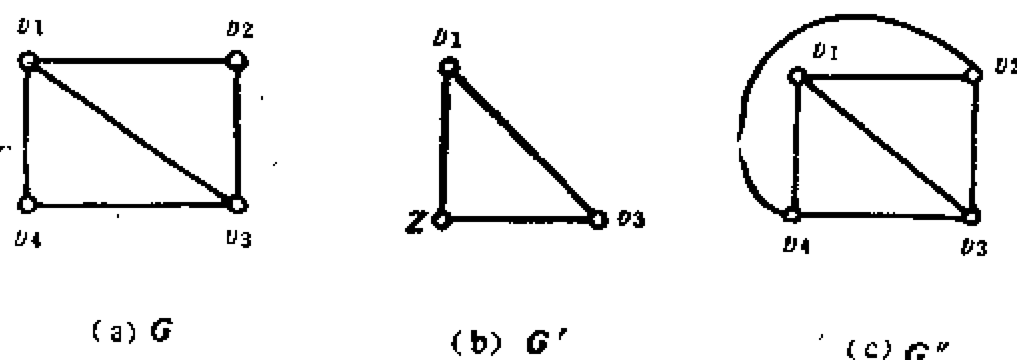


图 6.19

定理 6.18 设v_i和v_j是图G的两个不相邻结点，则G的点色数为

$$\psi_v(G)=\min\{\psi_v(G'),\ \psi_v(G'')\} \tag{6.17}$$

证：对图G的点着色时，v_i和v_j的颜色有两种可能，即v_i与v_j同色或v_i与v_j不同色。

若v_i与v_j同色，则凝缩后z点可用同一颜色而不须改变其他结点的颜色，故$\psi_v(G)=\psi_v(G')$，但并接时v_i和v_j须着不同颜色，则$\psi_v(G'')\geqslant\psi_v(G)$

若v_i与v_j不同色，则并接后不须改变v_i和v_j的颜色，即$\psi_v(G'')=\psi_v(G)$，但原来与v邻接的点可能有与v_j相同的颜色，而原来与v_j邻接的点可能有与v_i相同的颜色，因此当v_i和v_j凝缩成z点后，可能须要着既不同于v_i也不同于v_j的另一种颜色，因此$\psi_v(G')\geqslant\psi_v(G)$。

综合两种情况都可得到

$$\psi_v(G)=\min\{\psi_v(G'),\ \psi_v(G'')\}$$ ▮

应用定理6.18，反复对图进行并接和凝缩，使图的阶逐次变小，最后变成完全图，其中阶数最小的完全图，它的色数（也就是完全图的阶数）即为G的色数。

例 6.4 求图6.20中图G的点色数。

解：凝缩和并接过程如图，其中实心园点表示所考虑的两个不邻接点v_i和v_j。

由图解过程可见，最后得到的都是一些完全图，最小的是K_3，故$\psi_v(G)=3$。

二、独立集法

由于独立集内的所有结点可着同一种颜色，如果将图的结点进行划分，使每一结点子集都是独立集，则最小划分数即是图的点色数。

为此，先求图G的一个最大独立集V_1，然后求$G-V_1$的最大独立集V_2，再求$G-(V_1\cup V_2)$的最大独立集V_3，如此继续下去直到最后一个最大独立集V_k，则$\psi_v(G)=K$。

例 6.5 求图6.21的图的点色数。

解：先求G的最大独立集V_1，为此用布尔算法

$$B=\prod_{(u,\ v)\in E}(u+v)$$

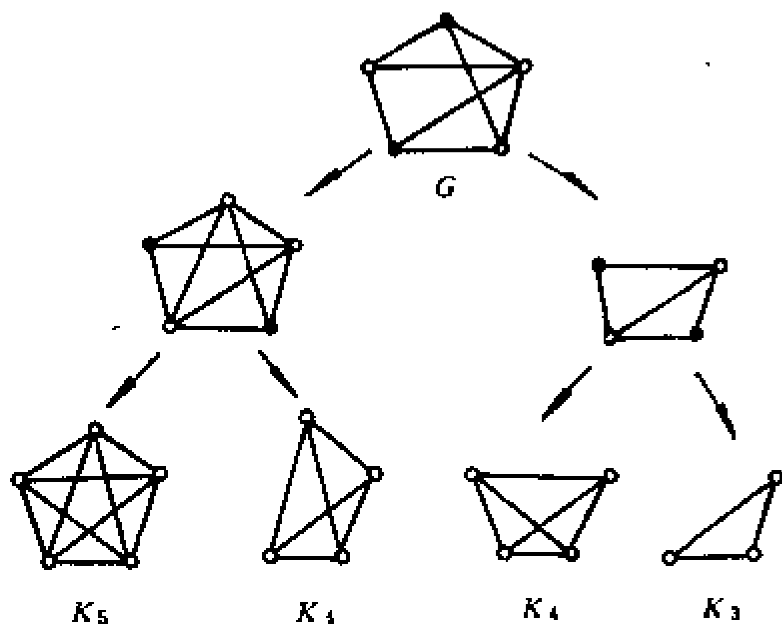


图 6.20

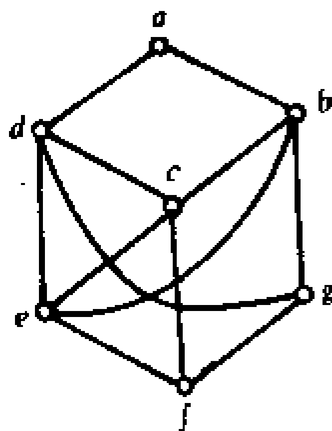


图 6.21

$$=(a+b)(a+d)(b+c)(b+e)\times$$
$$\times(b+g)(c+d)(c+e)(c+f)(d+e)(d+g)(e+f)$$
$$\times(f+g)$$
$$aceg+bdef+bdcf+bdceg$$

于是得到4个极大独立集：

$\{b, d, f\}$，$\{a, c, g\}$，$\{a, e, g\}$，$\{a, f\}$可以看到，其中$\{b, d, f\}$和$\{a, c, g\}$是两个不相交的最大独立集，可以作为V_1和V_2，剩下一个结点e独自构成一个独立集，于是得到图G的最小划分为

$$V_1=\{b, d, f\}$$
$$V_2=\{a, c, g\}$$
$$V_3=\{e\}$$

每一独立集的结点着相同颜色，故$\psi_v(G)=3$。

三、鲍威尔（Powell）法

步骤如下：

（1）将图G的结点按次数递减的顺序进行排列。（次数相同的结点之间的顺序可任意选定）。

（2）以第一种颜色给第一个结点着色，并按顺序，对与前面已着色的结点不相邻的结点着上同样的颜色。

（3）用第二种颜色对尚未着色的结点重复（2），继而用第三种颜色重复进行、直到所有结点都已着色为止。

例 6.6 试用鲍威尔法对图6.21的图着色

解：将结点按次数递减的顺序排列如下：

c, e, b, d, f, g, a

用颜色c_1对结点c着色，并顺序用同样颜色对结点g和a着色。

用颜色c_2对结点e着色，并顺序检查后面结点，因b，d，f都与e邻接，而g，a已着色，故C_2只对点e着色。

用颜色c_3对结点b着色，并顺序对结点d和f用同样颜色着色。

至此，全部结点着色完毕，所以$\varphi_v(G)=3$。

四、顺序着色法

将图的n个结点从1到n排序，记$A(v)$为与v邻接的结点集合，算法中采用了一布尔变量$N_v[j]$，当与结点v邻接的某一结点已着上色j时，$N_v[j]$即为真，对任一结点v，$N_v[j]$的初始值均为假。算法如下：

```
1. for i=1 to n do
       begin
2. While N_{v_i}[j] do j←j+1
3. for all i∈A(v_i) do N_v[j]←true
4. C(v_i)←j
   end
```

这一算法的时间复杂性为$O(n_2)$，然而，算法受结点排列次序的影响极大，即结点按不同的次序排列，算法得到的结果相差很大。例如偶图$G=(V, E)$，V的两个结点子集为$V_1=\{v_1, v_2, \cdots, v_k\}$，$V_2=\{u_1, u_2, \cdots, u_k\}$，设$E=\{v_i, u_j)|i\neq j\}$，如果结点按下面次序排列：

$$v_1, v_2, \cdots v_k, u_1, u_2, \cdots, u_k$$

则图只须最少的颜色数着色（2色），但是如果结点按以下次序排列：

$$v_1, u_1, v_2, u_2, \cdots, v_k, u_k$$

则这一算法需要的颜色数$K=\frac{1}{2}n$。显然，当n很大时，这种算法的效果是很差的。

§6.6 着色多项式

上一节我们讨论了图$G=(V, E)$的点色数$\varphi_v(G)$，即在保证正常着色的条件下结点着色至少需要多少种颜色。这一节我们讨论结点着色的方案，即在保证正常着色的条件下，如何把不同的颜色分配给各个结点，或者说使各点着色有多少种不同的分配方案。

如图6.22是3个结点的完全图，显然色数$\varphi_v(G)=3$，即至少需要3种颜色。假设现在有红。黄、兰三种颜色，那么这三种颜色如何分配给三个结点呢？可以有6种不同的着色方案，即

1. v_1←红色，v_2←黄色，v_3←蓝色
2. v_1←红色，v_2←蓝色，v_3←黄色
3. v_1←黄色，v_2←红色，v_3←蓝色
4. v_1←黄色，v_2←蓝色，v_3←红色
5. v_1←蓝色，v_2←红色，v_3←黄色
6. v_1←蓝色，v_2←黄色，v_3←红色

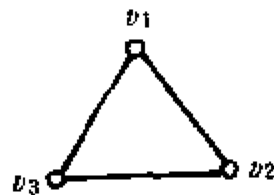


图 6.22

现在如果有K种颜色（$K\geqslant\varphi_i(G)$），那么上面的着色方案又有多少种呢?首先考虑结点v_1，在K种颜色中可任选一种，因此v_1有K种不同的着色方案，v_1着色后，剩下的$K-1$种颜色，可以任选一种给v_2着色，因此v_2有（$K-1$）种着色方案，同理，当v_1，v_2各着一种颜色之后，v_3有（$K-2$）种不同选择，因此总的着色方案数为：$K(K-1)(K-2)$。

着色方案最早是由 Birkhoff (1912) 作为攻克四色猜想的一种可能的手段提出来的。我们用 $P_k(G)$ 表示图 G 的不同的 K 着色的数目，下面将会看到，它是一个 K 的多项式。

定义 6.8 设图 G 有 n 个结点，k 的多项式 $P_k(G)$ 的值给出颜色数不超过 k 时给图的结点着色的不同方案数。称 $P_k(G)$ 为图 G 的着色多项式。

设 m_i 表示用 i 种颜色对图的结点着色的不同方案数，现有 k $(k \geqslant i)$ 种颜色对图 G 进行着色，则从 k 种颜色中取出 i 种颜色共有 $\binom{k}{i}$ 种不同取法，因此从 k 种颜色中取出 i 种颜色对图 G 着色总的不同方案数为 $m_i\binom{k}{i}$。故

$$P_k(G)=m_1\binom{k}{1}+m_2\binom{k}{2}+\cdots m_n\binom{k}{n}$$

$$=m_1k+\frac{m_2}{2!}k(k-1)+\frac{m_3}{3!}k(k-1)(k-2)+\cdots$$

$$+\frac{m_n}{n!}k(k-1)(k-2)\cdots(k-n+1) \tag{6.18}$$

故 $P_k(G)$ 为 k 的 n 次多项式。

例 6.7 设用 5 种颜色对图 6.23 的图的结点正常着色，问共有多少种不同的着色方案。

解：由着色多项式得

$$P_k(G)=m_1k+\frac{m_2}{2!}k(k-1)+\frac{m_3}{3!}k(k-1)(k-2)$$

$$+\frac{m_4}{4!}k(k-1)(k-2)(k-3)$$

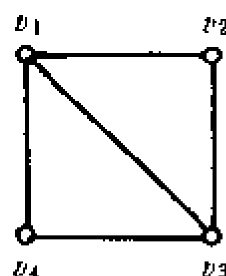


图 6.23

因为图 G 包含 k_3，至少需要 3 种颜色方能正常着色，故 $m_1=m_2=0$。

考虑 m_3，即用 3 种颜色的着色方案数，由于结点 v_1，v_2 和 v_3 必须着不同颜色，因此三种颜色已用尽，故 v_4 只能与 v_2 同色，v_1，v_2，v_3 着三种颜色的方案有 6 种，即 $m_3=6$。同理可知 $m_4=4!$，故

$$P_k(G)=k(K-1)(k-2)+k(k-1)(k-2)(k-3)$$

将 $k=5$ 代入即得总的方案数为

$$P_5(G)=5\cdot4\cdot3+5\cdot4\cdot3\cdot2=180$$

下面给出一些特殊图的着色多项式。

1. 零图 G_0

因为零图 G_0 的 n 个结点都是孤点，所以每一结点均可独立地从 k 种颜色中任选一种着色，故

$$P_k(G_0)=k^n \tag{6.19}$$

2. 完全图 K_n

正常着色的条件为 $k \geqslant n$，此时第一个结点可以有 k 种颜色的选择，第二个结点可以有 $k-1$ 种颜色的选择，第三个结点可以有 $k-2$ 种颜色选择，依此类推，第 n 个结点可以有 $K-n+1$ 种颜色的选择，故

$$P_k(K_n)=k(k-1)(k-2)\cdots(k-n+1) \tag{6.20}$$

3. 树 T_n

$$P_k(T_n)=k(k-1)^{n-1} \tag{6.21}$$

式（6.21）的证明留作习题。

下面讨论一般图的着色多项式。

上一节曾经讲过图的凝缩和并接，设图 $G=(V,E)$，G 的凝缩图为 G'，G 的并接图为 G''，G 的着色多项式记作 $P_k(G)$，而 G' 和 G'' 的着色多项式分别记作 $P_k(G')$ 和 $P_k(G'')$，它们之间有如下关系。

定理 6.19 若 v_i 和 v_j 是图 G 的不相邻二结点，则

$$P_k(G)=P_k(G')+P_k(G'') \tag{6.22}$$

证： 用 k 种颜色对 G 的结点进行着色，方案可分为两大类，一类是 v_i 和 v_j 着同一颜色，这一类方案的数目与用 k 种颜色对图 G' 的结点着色的方案数目相同。另一类是 v_i 和 v_j 着不同颜色，这一类方案的数目与用 k 种颜色对图 G'' 的结点着色的数目，这两类方案数目之和，就是用 k 种颜色对图 G 的结点进行着色总的方案数。即

$$P_K(G)=P_K(G')+P_K(G'')$$ ▌

定理6.19为我们提供了一个求任意图着色多项式较为简便的递推方法，就是将图不断地并接和凝缩，最后得到若干个完全图，则原图的着色多项式，等于所有这些完全图着色多项式之和。

例 6.8 求图 6.24 中图 G 的着色多项式。

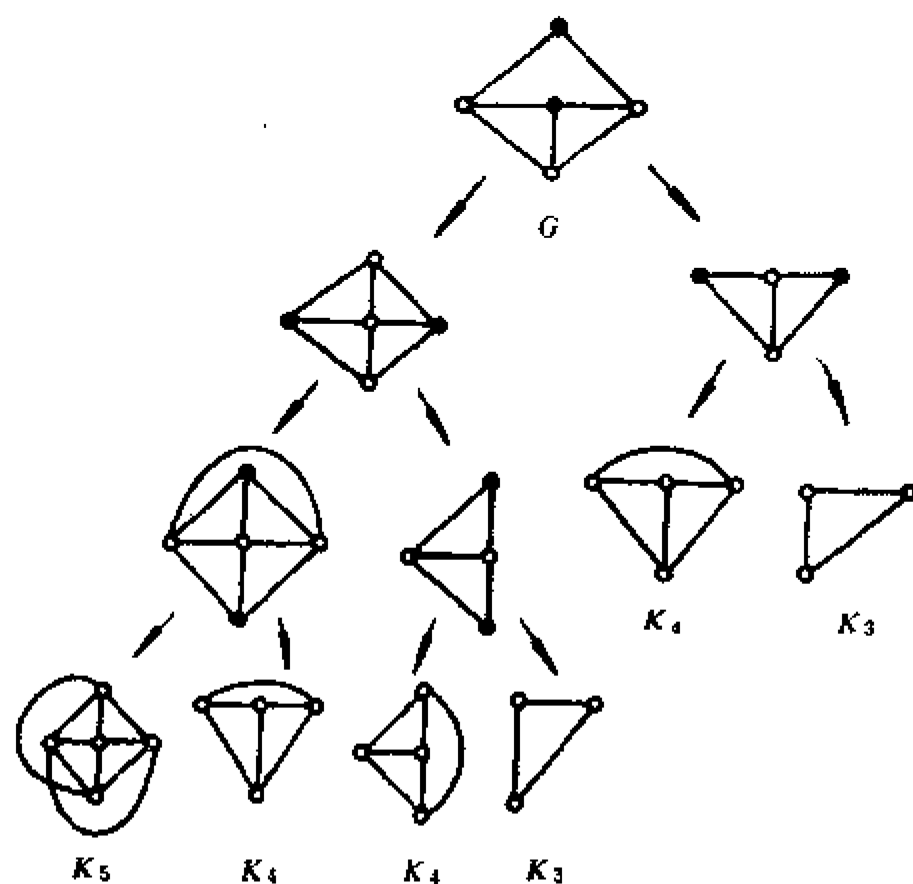


图 6.24

解： 凝缩和并接过程如图所示，其中实心圆点表示两个所考虑的不相邻结点 v_i 和 v_j。由图解过程可知，最后得到一个 K_5，3 个 K_4 和两个 K_3，因此

$$P_K(G)=P_K(K_5)+3\cdot P_K(K_4)+2\cdot P_K(K_3)$$

根据式（6.20）即得

$$P_K(G)=k(k-1)(k-2)(k-3)(k-4)+3k(k-1)(k-2)\times(k-3)+2k(k-1)(k-2)$$

$$=k(k-1)(k-2)(k^2-4k+5)$$

如果将（6.22）式改写成下式

$$P_K(G'')=P_K(G)-P_K(G') \tag{6.23}$$

G''是v_i和v_j之间已有一条连接边 $e=(v_i,\ v_j)$ 的图，而G相当于在 G''中去掉边 e 后得到的图，G' 则是在G''中将边 e 的两个端点凝缩为一点（或将边 e 收缩成一点）得到的图，等式给出 G''的着色多项式等于G与 G' 着色多项式之差。因此，我们可以应用这个公式求任意一个图的着色多项式。方法是去掉图的任意一条边，另一方面又将这条边收缩成一点，于是得到两个图，对形成的图重复上述过程，边的数目将逐渐减少，最后得到的都是不含任何边的零图，根据（6.19）和（6.23）式，即可得到原图的着色多项式。

例 6.9 求图 6.25 中图G的着色多项式。

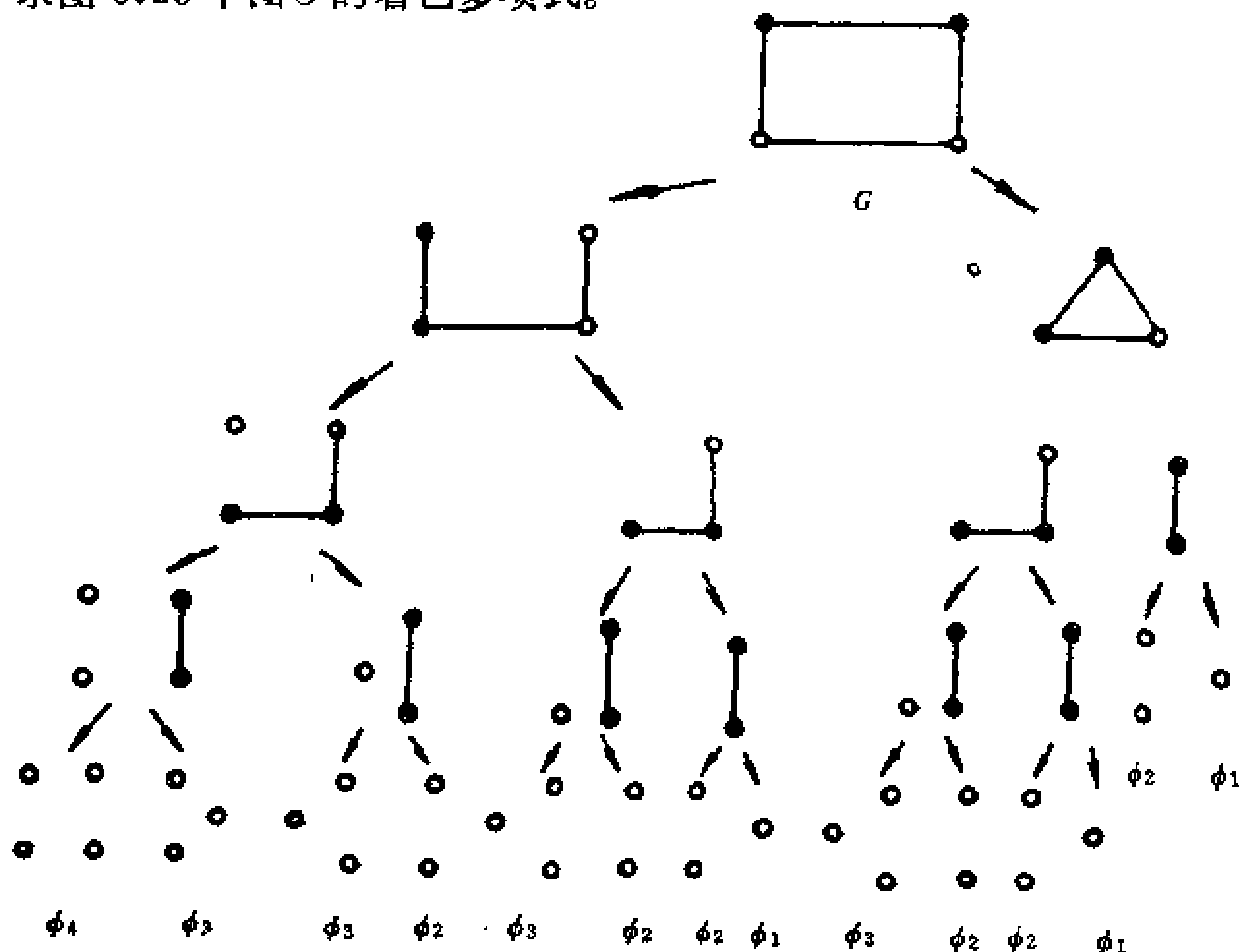


图 6.25

解：边的删除和收缩过程如图所示，其中实心圆点之间的边即为所操作的边。

用 ϕ_i 表示有 i 个孤点的零图，由上图可得

$$P_K(G)=P_K(\phi_4)-4P_K(\phi_3)+6P_K(\phi_2)-3P_K(\phi_1)$$
$$=k(k-1)(k^2-3k+3)$$

应用公式（6.22）的递推方法是使图的边数逐渐增多，最终成为完全图，而利用公式（6.23）的递推方法是使图的边数逐渐减少，最终成为零图，所以当图的边数较多时适于前一公式，当图的边数较少时用后一公式比较适合。

习题与思考题

1. 试用三种颜色，给图 6.26 所示的图的面正常着色。
2. 给图 6.27 的三个图的结点正常着色，每个图至少需要几种颜色。

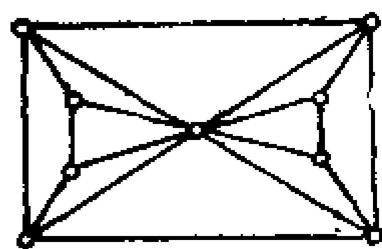
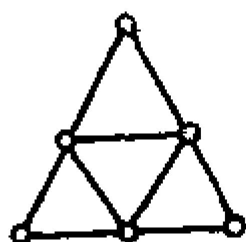
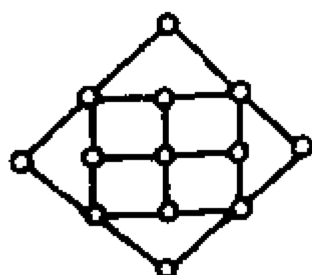


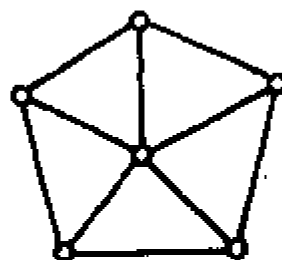
图 6.26



(a)



(b)



(c)

图 6.27

*3．证明：若G是一个至少有4个结点的3可着色平面图，则G至少含有二个三角形。

4．证明：彼特森图（图4.23(a)）是边4色的。

5．证明：若G是无环图且$\Delta=3$，则

$$\psi_e(G)\leqslant 4$$

6．证明：若$G=(V,E)$是非空的正则简单图，且$|V|$是奇数，则$\psi_e(G)=\Delta+1$

7．证明：若$G=(V,E)$是无环图，且$|V|=2k+1$及$|E|>k\Delta$，则$\psi_e(G)=\Delta+1$

8．在一所学校里，有七位教师和十二个班级，五天一周的教学任务由下面的矩阵给出，这里P_{ij}的值表示教师x_i给班级y_j的授课时数。问

（a）一天必须安排多少课时方能满足要求。

（b）若一张每天8个课时的课表已经排出，则需要多少教室。

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}
x_1	3	2	3	3	3	3	3	3	3	3	3	3
x_2	1	3	6	0	4	2	5	1	3	3	0	4
x_3	5	0	5	5	0	0	5	0	5	0	5	5
x_4	2	4	2	4	2	4	2	4	2	4	2	3
x_5	3	5	2	2	0	3	1	4	4	3	2	5
x_6	5	5	0	0	5	5	0	5	0	5	5	0
x_7	0	3	4	3	4	3	4	3	4	3	3	0

9．试用布尔算法求图6.12的图的极大独立集和$I(G)$。

10．设图G是n阶简单图，证明：

$$I(G)+\psi_v(G)\leqslant n+1$$

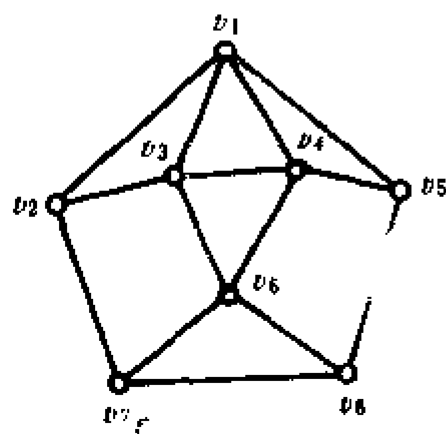


图 6.28

11．试证：9个结点17条边的平面图不可能用两种颜色对图的面正常着色。

12．试用布尔算法求图6.28所示的图的极小支配集及$D(G)$。

13．证明：在偶图中，最大匹配的边数等于最小复盖的结点数。

14．设P是简单图G的某一完全j分图，$T_{j,n}$为有n个结点的完全j分图，即它的每部分或是有$\left\lfloor \frac{n}{j} \right\rfloor$个结点，或是有$\left\lceil \frac{n}{j} \right\rceil$个结点，试证明：

$$|E(P)| \leqslant |E(T_{j;n})|$$

并且仅当$G \cong T_{j,n}$时，等式成立。

15．设$G=(V,E)$，$|V|=n$，$|E|=m$，试证明：若G是简单图，则

$$\phi_v(G) \geqslant n^2/(n^2-2m)$$

16．证明：若G的任意两个奇次回路都有一公共结点，则$\phi_v(G) \leqslant 5$。

17．若G的任意真子图H均有$\phi_v(H) < \phi_v(G)$，则称G为临界图，此时若$\phi_v(G)=k$，则称G为k临界子图。试举一4临界图的例子。

18．设δ表示图结点的最小次数，试证明：若G是K临界图，则

$$\delta \leqslant K-1$$

19．证明：每个结点K色图至少有K个次数不小于$K-1$的结点。

20．设d_i表示结点v_i的次数，证明：若图G有结点次序列$(d_1, d_2, \cdots, d_n)$，且$d_1 \geqslant d_2 \geqslant \cdots d_n$，则

$$\phi_v(G) \leqslant \max \min\{d_i+1, i\}$$

21．利用上题结果证明：当$|E|>0$时

$$\phi_v(G) \leqq \lceil (2|E|)^{\frac{1}{2}} \rceil$$

22．证明：$\phi_v[J(G_1, G_2)] = \phi_v(G_1) + \phi_v(G_2)$

*23．应用定理6.17（Brooks）证明：若G是$\Delta=3$的无环图，则$\phi_e(G) \leqslant 4$

24．在九个人的人群中，有一个人认识另外两个人，有两个人每人认识另外四个人，有四个人每人认识另外五个人，余下的两个人每人认识另外六个人。证明：有三个人他们彼此都互相认识。

25．试用凝缩、并接法求图6.29的图G的点色数。

26．试用独立集法求图6.29图的点色数。

27．试用鲍威尔法对图6.30的图的结点着色。

28．试用顺序着色法对图6.30的图的结点着色。

29．试将图6.31的边着色问题转化为点着色问题求解。

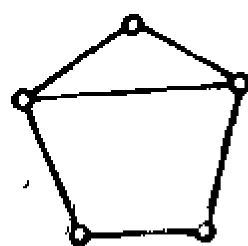


图 6.29

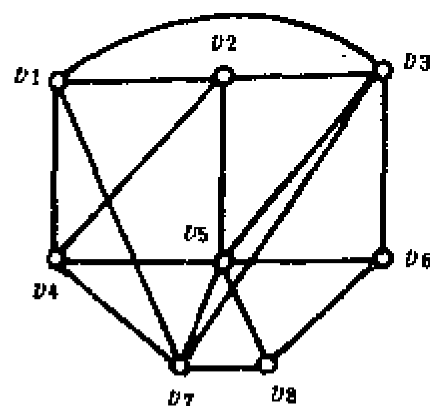


图 6.30

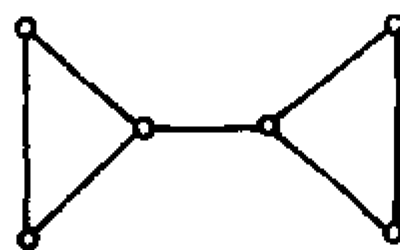


图 6.31

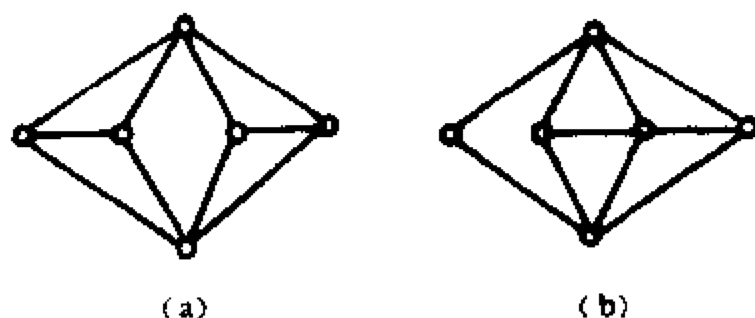


图 6.32

30．证明：n 个结点的树的着色多项式为：$P_k(T_n)=k(k-1)^{n-1}$。

31．试用公式（6.22）计算图 6.32（a）和（b）两个图的着色多项式。

32．试用公式（6.23）的算法重复上题。

33．证明：n 个结点的连通图的多项式满足下列不等式：

$$P_K(G)\leq k(k-1)^{n-1}$$

34．试证：不存在以 $k^4-3k^3+3k^2$ 为其着色多项式的图。

第七章 路径问题

路径问题是图论的一个很广泛的问题，也是图论在应用领域中一项卓有成效的内容。当我们在处理各种实际问题时，我们将给出各种各样的图的数学模型，例如交通运输图、通讯线路图、决策图、工程规划图等等，在研究这些图时，除了考虑连通性和可达性之外，另一重要问题就是考虑最短路径、最优路径和关键路径。这一章我们分别讨论达些路径的应用及其算法。

§7.1 从指定点到其他点的最短路径

定义 7.1 在带权有向图D中，弧$\langle v_i, v_j\rangle$上带的权$w(v_i, v_j)$称为这条弧的长度。一条路径的长度是这条路径所经过的弧的长度和。即若P是D中的一条路径，则P的长度L_P为：

$$L_P=\sum_{\langle u,v\rangle\in P} w(u,v)$$

定义 7.2 在带权图D中，若P是从结点u到结点v的所有路径中最短的一条路径，则称P为从u到v的最短路径，其长度记作$L(u,v)$并称为从u到v的距离。

最短路径具有重要的实际意义。例如用结点表示城市，弧的长度表示一个城市到另一城市的公路里程，则最短路径问题就是寻找从一城市到另一城市的最短里程问题，如果弧的长度表示修筑道路的经费，则最短路径表示最小投资、等等。

上述最短路径的定义可以推广到带权无向图中。

怎样寻找从某一指定点到其他点的最短路径呢？1959年得克斯特拉（Dijkstra）提出了一个算法，是当前普遍采用的一个较好算法，这个算法的主要思路是用逐点增长的方法构造一棵路径树，从而得到从树根（指定点）到其他点的距离。下面先介绍这一思路。

设求从指定点v_0到某一点v的最短路径。

从v_0出发沿着弧的方向第一个到达而且路径最短的一点，一定是所有与v_0有弧关联且弧的长度最小的那个点，这是与v_0距离最短的一点，称为第一短路径点，如果这个点就是v，问题得到解决，如果不是v而是另一点v_1，那么从v_0出发第二条短路径会到达哪个点呢？一定是与v_0有弧关联且弧的长度最小的那个点（v_1除外），或者是由v_0经过v_1再经过与v_1关联的长度最小的一条弧所到达的点，取这两条路径中最短的一条，它的终点即是从v_0出发第二条短路径所到这的点，设为v_2，如果v_2即为所求的v，问题即得解决，否则继续寻求从v_0出发第三短路径到达的点，第四短路径到达的点，如此继续下去直到路径的终点是v为止。

一般说来，设T为已求得的第一、第二，…第k条短路径的终点集合，即$T=\{v_1, v_2, \cdots, v_k\}$，则有$L(v_0, v_1)\leqslant L(v_0, v_2)\leqslant\cdots\leqslant L(v_0, v_k)$。那么第$k+1$条短路径（设其终点为$w$）或者是弧$\langle v_0, w\rangle$，或者终点是$T$外的某一点$w$，但路径的中间结点一定都是

T 中的结点（即已到达过的结点）这是显然的，如果从 v_0 到 w 的路径中有一中间点 i 不在 T 中，由于 $L(v_0,i)<L(v_0,w)$ 则 i 点必然作为比 v_0 到 w 的路径更短的一条路径终点先行加入 T 中。因此按照这种路径逐步增长的过程，我们将得到一棵以 v_0 为根的树，称为路径树，如图 7.1所示，从树根到各结点的路径长度即是 v_0 到该点的距离。

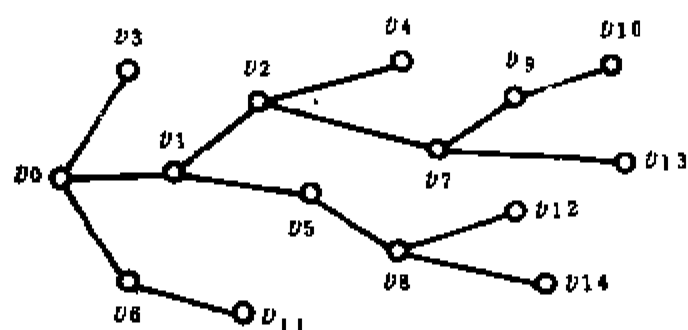


图 7.1

根据以上分析我们可以得出求路径端点的算法。

设已求出从 v_0 出发长度依次递增的 k 条路径，它们的端点各为 v_1，v_2，…，v_k，并归入已到达的端点集合 T 中，并将 v_0 到 v_i 的距离简记为 $L(v_i)$，对 T 外的任一结点 v'，计算数值

$$L(v')=\min_{0\leqslant i\leqslant k}\{L(v_i)+w(v_i,\ v')\} \qquad (A)$$

式中约定 $L(v_0)=0$，若 $(v_i,\ v')\notin E$，则令 $w(v_i,\ v')=\infty$（在计算机中则用某一允许的最大值代替 ∞）。

因此，第 $k+1$ 条短路径，其端点一定是 T 外各点中 $L(v)$ 最小的那个点（这里已将 v_0 加入 T 中），设为点 v_{k+1}，则

$$L(v_{k+1})=\min\{L(v')\mid v'\notin T\} \qquad (B)$$

式（A）和（B）给出了求第 $k+1$ 条短路径终点的公式，然而直接应用（A）式计算是很麻烦的，因为每求一条路径的终点，都要反复地对已通过的结点进行计算和比较，我们可以加以改进，即在求最短路径的过程中，不断修改未通过点的路径参数，于是得到如下算法：

Dijkstra 最短路径算法：

1. for all $v\neq u$ $L(v)\leftarrow w(u,\ v)$
2. $L(u)\leftarrow 0$
3. $T\leftarrow\{u\}$
4. While $T\neq V$ do
   begin
5. 找一点 $v'\notin T$ 且对所有 $v\notin T$ $L(v')\leqslant L(v)$
6. $T\leftarrow T\cup\{v'\}$
7. for all $v\notin T$
   if $l(v)>L(v')+w(v',v)$
   then $L(v)\leftarrow L(v')+w(v',v)$
   end

算法说明：

算法中指定结点设为 u，$w(u,u)=0$，T 为通过的结点集合，初始值为 $\{u\}$，当 $T=V$ 时表示图的所有结点已通过，计算结束。计算从第 4 行循环语句开始，在不属于 T 的结点中选路径长度最小的一点 v'，将 v' 加入 T 中并根据 v' 修改不属于 T 的结点的路径长度，如此反复进行直到所有结点都加入 T 中为止。因此这一算法是求从指定点 u 到所有其余结点的距离，其值由通过结点 v' 的 $L(v')$ 给出。如果只需要求 u 到某些点的距离，那么只要

对第4行的循环语句稍作修改即可达到目的。这一算法的时间复杂性主要耗费在第4行开始的循环体中，而循环次数不会超过 $(n-1)$，在循环体中，第5行和第7行的执行时间是 $O(n)$ 级的，总的运算时间是 $O(n^2)$ 级的。

例 7.1 求图7.2中结点 u 到其余各点的距离。

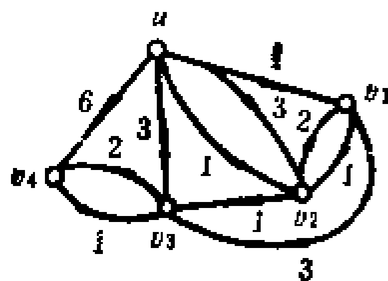


图 7.2

解：由图得出初始值：

$$L(v_1)=1,\ L(v_2)=3,$$

$$L(v_3)=\infty,\ L(v_4)=6,\ L(u)=0,\ T=\{u\}$$

1）选择 T 外路径最小一点 v_1，于是有

$$T=\{u,\ v_1\}$$

根据 v_1 修改 T 外各点路径

因 $$L(v_1)+w(v_1,\ v_2)=1+1<L(v_2)=3$$

故 $$L(v_2)\longleftarrow 2,$$

同理 $$L(v_3)\longleftarrow 1+3=4,\ L(v_4)=6\ (\text{不改变})$$

2）选择 T 外路径最小一点 v_2，于是有

$$T=\{u,\ v_1,\ v_2\}$$

根据 v_2 修改 T 外各点路径

因 $$L(v_2)+w(v_2,\ v_3)=2+1<L(v_3)=4$$

故 $$L(v_3)\longleftarrow 3$$

而 $$L(v_4)=6\ (\text{不改变})$$

3）选择 T 外路径最小一点 v_3，于是有

$$T=\{u,\ v_1,\ v_2,\ v_3\}$$

根据 v_3 修改 T 外各点路径，

$$L(v_4)\longleftarrow 3+2=5$$

4）得到最后一点 v_4，于是有 $T=\{u,\ v_1,\ v_2,\ v_3,\ v_4\}$

最后 $T=V$，计算结束。u 到各点的距离为：$L(v_1)=1$，$L(v_2)=2$，$L(v_3)=3$，$L(v_4)=5$。

得克斯特拉算法只给出指定点到其余结点的距离，并未直接给出指定点到其余各点的最短路径，为此，可以对上述算法稍加补充，即增设一路径变量 $P(v_i)$，其初始值规定如下

$$P(v_i)=\begin{cases}\{u\}\cup\{v_i\} & \text{若 } w(u,\ v_i)\neq\infty \\ \phi & \text{若 } w(u,\ v_i)=\infty\end{cases}$$

经过各次迭代和修改过程，最后得到的 $P(v_i)$ 和 $L(v_i)$ 即为指定点 u 到点 v_i 的最短路径和距离。算法如下。

1. for all $v\in V$，给出 $L(v)$ 和 $P(v)$ 的初始值
2. $T\longleftarrow\{u\}$
3. While $T\neq V$ *do*

   begin

4. 找一点 $v'\notin T$ 且对所有 $v\notin T$ 有 $L(v')\leqslant L(v)$
5. $T\longleftarrow T\cup\{v'\}$

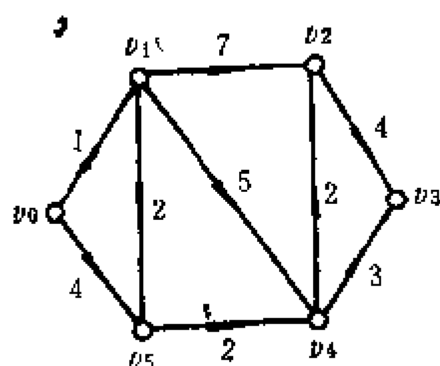


图 7.3

6. for all $v \notin T$

  if $L(v) > L(v') + w(v', v)$

  then $L(v) \leftarrow L(v') + w(v', v)$

  $P(v) \longleftarrow P(v') \cup \{v\}$

  end

例 7.2 求图 7.3 中结点v_0到其余各点的最短路径和距离。

解：由图得出各结点的$L(v)$和$P(v)$的初始值，计算的迭代过程列成表如下，表中v'

迭代次数	v'	v_0		v_1		v_2		v_3		v_4		v_5		T
		$L(v_0)$	$P(v_0)$	$L(v_1)$	$P(v_1)$	$L(v_2)$	$P(v_2)$	$L(v_3)$	$P(v_3)$	$L(v_4)$	$P(v_4)$	$L(v_5)$	$P(v_5)$	
0	/	0	$\{v_0\}$	1	$\{v_0, v_1\}$	∞	ϕ	∞	ϕ	∞	ϕ	4	$\{v_0, v_5\}$	$\{v_0\}$
1	v_1			1	$\{v_0, v_1\}$	8	$\{v_0, v_1, v_2\}$	∞	ϕ	6	$\{v_0, v_1, v_4\}$	3	$\{v_0, v_1, v_5\}$	$\{v_0, v_1\}$
2	v_5					8	$\{v_0, v_1, v_2\}$	∞	ϕ	5	$\{v_0, v_1, v_5, v_4\}$	3	$\{v_0, v_1, v_5\}$	$\{v_0, v_1, v_5\}$
3	v_4					7	$\{v_0, v_1, v_5, v_4, v_2\}$	8	$\{v_0, v_1, v_5, v_4, v_3\}$	5	$\{v_0, v_1, v_5, v_4\}$			$\{v_0, v_1, v_5, v_4\}$
4	v_2					7	$\{v_0, v_1, v_5, v_4, v_2\}$	8	$\{v_0, v_1, v_5, v_4, v_3\}$					$\{v_0, v_1, v_5, v_4, v_2\}$
5	v_3							8	$\{v_0, v_1, v_5, v_4, v_3\}$					$\{v_0, v_1, v_5, v_4, v_2, v_3\}$
6	$T = V$ 停止													

即为每次访问的结点，该行的$P(v')$和$L(v')$即为v_0到该点的最短路径和距离。

上面讲的是带权有向图，如果在算法中用边代替弧，算法同样适用于带权无向图的情形。最后应指出，得克斯特拉算法只适合于权值为非负实数的情形，如果边权的值有正有负，则算法不能保证所得到的一定是一条最短路径，1964 年，福特(Ford)提出了适用于负长度的弧的最短路径的算法，限于篇幅这里就不介绍了，有兴趣的读者可参阅有关文献〔5〕

§7.2 任意两点间的最短路径

在第二章我们已经给出了带权图邻接矩阵的定义，可以利用带权图的邻接矩阵求图的任意两点之间的距离。为此先定义矩阵的两种运算。

定义 7.3 设$A=(a_{ij})_{n\times k}$，$B=(b_{ij})_{k\times m}$，定义$C=A*B=(C_{ij})_{n\times m}$，其中

$$C_{ij}=\min\{a_{i1}+b_{1j},\ a_{i2}+b_{2j},\ \cdots,\ a_{ik}+b_{kj}\} \tag{7.1}$$

定义 7.4 设$A=(a_{ij})_{n\times m}$，$b=(b_{ij})_{n\times m}$，定义$Q=A\circledast B=(q_{ij})_{n\times m}$，其中

$$q_{ij}=\min\{a_{ij},\ b_{ij}\} \tag{7.2}$$

现在我们来看邻接矩阵应用这两种运算所产生的结果。先讨论带权无向图的情形。

设$A=(a_{ij})_{n\times n}$是带权无向图$G=(V,\ E)$的邻接矩阵，记$A^{(2)}=A*A=(a_{ij}^{(2)})_{n\times n}$，则

$$a_{ij}^{(2)}=\min\{a_{i1}+a_{1j},\ a_{i2}+a_{2j},\ \cdots,\ a_{in}+a_{nj}\}$$

这里$a_{i1}+a_{1j}$表示从结点i经过中间点1到结点j的路径长度，$a_{i2}+a_{2j}$表示从结点i经过中间点2到结点j的路径长度，其余项的意义相同，都表示从结点i经过一个中间点到结点j的路径长度，$a_{ij}^{(2)}$是取它们中的最小值，它的意义就是从结点i最多经过一个中间点到结点j的所有路径中最短的那条路径。

所以说最多经过一个中间点是因为一般项$a_{ik}+a_{kj}$（$1\leqslant k\leqslant n$）中也包含$a_{ii}+a_{ij}$和$a_{ij}+a_{jj}$两项，它们都等于a_{ij}，如果它是所有项中的最小值，则$a_{ij}^{(2)}=a_{ij}$，在这种情况下就没有经过中间点。

同理可知，在$A^{(k)}=(a_{ij}^{(k)})_{n\times n}$中，$a_{ij}^{(k)}$表示从结点$i$最多经过（$k-1$）个中间点到结点$j$的所有路径中最短的那条路径。

因为图的阶数为n，从i到j的简单路径最多经过（$n-2$）个中间结点，所以只要求到$A^{(n-2)}$就行了，然后比较a_{ij}，$a_{ij}^{(2)}$，…，$a_{ij}^{(n-2)}$，取其中最小的一项则是从i到j的所有路径中长度最小的一条路径，即最短路径。

于是，得到求任意两点间距离的算法如下

1．给出图的邻接矩阵A

2．求出$A^{(2)}$，$A^{(3)}$，…，$A^{(n-2)}$

3．计算$D=A\circledast A^{(2)}\circledast\cdots\circledast A^{(n-2)}=(d_{ij})_{n\times n}$

　则D即为图的距离矩阵。

例 7.3 求图 7.4 所示的无向图的距离矩阵。

解：写出带权无向图的邻接矩阵如下

$$A=\begin{matrix} & a & b & c & d & e & f \\ a & 0 & 1 & \infty & 2 & \infty & \infty \\ b & 1 & 0 & 3 & 4 & \infty & \infty \\ c & \infty & 3 & 0 & 1 & 2 & 2 \\ d & 2 & 4 & 1 & 0 & 3 & \infty \\ e & \infty & \infty & 2 & 3 & 0 & 2 \\ f & \infty & \infty & 2 & \infty & 2 & \infty \end{matrix}$$

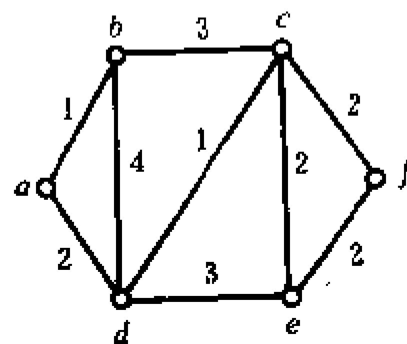


图 7.4

因为矩阵对称于主对角线，运算时只取下三角形。

先求$A^{(2)}$中的元素$a_{ij}^{(2)}$，例如：

$$\begin{aligned} a_{11}^{(2)}&=\min\{a_{11}+a_{11},\ a_{12}+a_{21},\ \cdots,\ a_{16}+a_{61}\}\\ &=\min\{0+0,\ 1+1,\ \cdots,\ \infty+\infty\}\\ &=0\\ a_{12}^{(2)}&=\min\{a_{11}+a_{12},\ a_{12}+a_{22},\ \cdots,\ a_{16}+a_{62}\}\\ &=\min\{0+1,\ 1+0,\ \cdots,\ \infty+\infty\} \end{aligned}$$

$$=1$$

$$\vdots$$

$$a_{34}^{(2)}=\min\{a_{31}+a_{14},\ a_{32}+a_{24},\ a_{33}+a_{34},\ \cdots,\ a_{36}+a_{64}\}$$
$$=\min\{\infty+2,\ 3+4,\ 0+1,\ \cdots,\ 2+\infty\}$$
$$=1$$

$$\vdots$$

于是求得 $A^{(2)}$，同理可求出 $A^{(3)}$和 $A^{(4)}$如下

$$A^{(2)}=\begin{array}{c}b\\c\\d\\e\\f\\\\\end{array}\begin{array}{c}\left[\begin{array}{ccccc}1&&&&\\3&3&&&\\2&3&1&&\\5&5&2&3&\\\infty&5&2&3&2\end{array}\right]\\\begin{array}{ccccc}a&b&c&d&e\end{array}\end{array}\quad A^{(3)}=\left[\begin{array}{ccccc}1&&&&\\3&3&&&\\2&3&1&&\\5&5&2&3&\\5&5&2&3&2\end{array}\right]\quad A^{(4)}=\left[\begin{array}{ccccc}1&&&&\\3&3&&&\\2&3&1&&\\5&5&2&3&\\5&5&2&3&2\end{array}\right]$$

令
$$D=A\circledast A^{(2)}\circledast A^{(3)}\circledast A^{(4)}$$
即得到图的距离矩阵如下

$$D=\begin{array}{c}b\\c\\d\\e\\f\\\\\end{array}\begin{array}{c}\left[\begin{array}{ccccc}1&&&&\\3&3&&&\\2&3&1&&\\5&5&2&3&\\5&5&2&3&2\end{array}\right]\\\begin{array}{ccccc}a&b&c&d&e\end{array}\end{array}$$

如果不仅需要求出任意两点之间的距离，还要求出两点之间的最短路径，那么在矩阵运算时要注以下标，并在运算中保留下标，即

$$a_{ik}+a_{kj}=a_{ikj}$$
$$a_{ijk}+a_{kmn}=a_{ijkmn}$$

如上例，图 7.4 的邻接矩阵应写成：

$$A=\begin{array}{c}b\\c\\d\\e\\f\\\\\end{array}\begin{array}{c}\left[\begin{array}{ccccc}1_{ba}&&&&\\\infty&3_{cb}&&&\\2_{da}&4_{db}&1_{dc}&&\\\infty&\infty&2_{ec}&3_{ed}&\\\infty&\infty&2_{fc}&\infty&2_{fe}\end{array}\right]\\\begin{array}{ccccc}a&b&c&d&e\end{array}\end{array}$$

矩阵中元素的下标表示结点，元素的值表示两个结点所关联的边的权值，如权为无限大可不写下标。经运算后，$a_{ij}^{(k)}$ 的下标表示路径，其值即表示这一路径的长度，例如求出 $A^{(2)}$如下：

$$A^{(2)}=\begin{array}{c}b\\c\\d\\e\\f\\\\\end{array}\begin{array}{c}\left[\begin{array}{ccccc}1_{ba}&&&&\\3_{cda}&3_{cb}&&&\\2_{da}&3_{dcb}&1_{dc}&&\\5_{eda}&5_{ecb}&2_{ec}&3_{ed}&\\\infty&5_{fcb}&2_{fc}&3_{fcd}&2_{fe}\end{array}\right]\\\begin{array}{ccccc}a&b&c&d&c\end{array}\end{array}$$

其中第4行第2列的元素按下式计算得到

$$\begin{aligned} a_{42}^{(2)} &= \min\{a_{41}+a_{12},\ a_{42}+a_{22},\ a_{43}+a_{32},\ \cdots,\ a_{46}+a_{62}\} \\ &= \min\{2_{da}+1_{ab},\ 4_{de}+0,\ 1_{dc}+3_{cb},\ \cdots,\ \infty+\infty\} \\ &= 3_{dab} \end{aligned}$$

其他幂矩阵的下标求法与此相同，最后即可求出带有下标的距离矩阵如下：

$$D = \begin{matrix} b \\ c \\ d \\ e \\ f \end{matrix}\begin{bmatrix} 1_{ba} & & & & \\ 3_{cda} & 3_{cb} & & & \\ 2_{da} & 3_{dab} & 1_{dc} & & \\ 5_{eda} & 5_{ecb} & 2_{ec} & 3_{ed} & \\ 5_{fcda} & 5_{fcb} & 2_{fc} & 3_{fcd} & 2_{fe} \end{bmatrix}$$
$$\qquad\quad a \quad\ b \quad\ c \quad\ d \quad\ e$$

从矩阵D中即可求出任意两点间的最短路径及其距离。例如：从结点f到a的最短路径为$fcda$，其距离为5。

上面的算法中，如果两点之间的最短路径不止一条，我们在写下标时只保留其中一条最短路径。

上面的例子是对无向图的，对于有向图，计算方法相同，但有两点必须注意：

（1）因无向图的邻接矩阵是对称的，可以只取上三角形或下三角形，对有向图不宜采用。

（2）对无向图，从v_i到v_j的最短路径，也是从v_j到v_i的最短路径，所以写下标时，我们把$acdf$与$fdca$当作一条路径，对有向图这是不可以的。

下面举一个有向图的计算例子。

例 7.4 如图7.5所示的带权有向图，试求任意两点之间的最短路径及距离。

解：先求出图的带权邻接矩阵如下

$$A = \begin{matrix} \\ a \\ b \\ c \end{matrix}\begin{matrix} a & b & c \\ \end{matrix}$$
$$A = \begin{matrix} a \\ b \\ c \end{matrix}\begin{bmatrix} 0 & 4_{ab} & 11_{ac} \\ 6_{ba} & 0 & 2_{bc} \\ 3_{ca} & \infty & 0 \end{bmatrix}$$

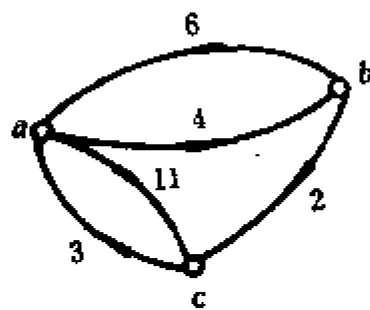


图 7.5

再求$A^{(2)}=(a_{ij}^{(2)})$，例如

$$\begin{aligned} a_{11}^{(2)} &= \min\{a_{11}+a_{11},\ a_{12}+a_{21},\ a_{13}+a_{31}\} \\ &= \min\{0+0,\ 4_{ab}+6_{ba},\ 11_{ac}+3_{ca}\} \\ &= 0 \end{aligned}$$

$$\begin{aligned} a_{12}^{(2)} &= \min\{a_{11}+a_{12},\ a_{12}+a_{22},\ a_{13}+a_{32}\} \\ &= 4_{ab} \end{aligned}$$

$$\begin{aligned} a_{13}^{(2)} &= \min\{a_{11}+a_{13},\ a_{12}+a_{23},\ a_{13}+a_{33}\} \\ &= 6_{abc} \end{aligned}$$

其余类似可求，于是求得$A^{(2)}$如下

$$A^{(2)} = \begin{matrix} a \\ b \\ c \end{matrix}\begin{bmatrix} 0 & 4_{ab} & 6_{abc} \\ 5_{bca} & 0 & 2_{bc} \\ 3_{ca} & 7_{cab} & 0 \end{bmatrix}$$
$$\qquad\qquad a \quad\ b \quad\ c$$

则距离矩阵为

$$D=A\circledast A^{(2)}=\begin{array}{c} \\ a \\ b \\ c \end{array}\begin{array}{c} \begin{matrix} a & b & c \end{matrix} \\ \begin{pmatrix} 0 & 4_{ab} & 6_{abc} \\ 5_{bca} & 0 & 2_{bc} \\ 3_{ca} & 7_{cab} & 0 \end{pmatrix} \end{array}$$

在求任意两点间最短路径的诸多算法中，弗洛伊得（Floyd，1962）提出的算法是一个较好的算法，现介绍如下：

算法的思路是直接在图的带权邻接矩阵中，用依次插入结点的方法改变矩阵的元素值，使最后得到的矩阵成为图的距离矩阵和路径矩阵。算法首先给图的结点编号（次序可以是任意的）并将邻接矩阵A作为距离矩阵D的初始值，即

$$D^{(0)}=A,\quad (d_{ij}^{(0)})_{n\times n}=(a_{ij})_{n\times n}$$

在矩阵D^0上，将结点v_1（标号为1）插入任意两点v_i与v_j（标号为i与j）之间，并按下式求出$D^{(1)}=(d_{ij}^{(1)})_{n\times n}$，

$$d_{ij}^{(1)}=\min\{d_{ij}^{(0)},\ d_{i1}^{(0)}+d_{1j}^{(0)}\}$$

可见$d_{ij}^{(1)}$是从v_i直接经过一条弧到v_j，或者从v_i经中间点v_1到v_j这两条路径中最短路径的长度。

在矩阵$D^{(1)}$上，将结点v_2（标号为2）插入任意两点v_i与v_j之间，并按下式求出$D^{(2)}=(d_{ij}^{(2)})_{n\times n}$

$$d_{ij}^{(2)}=\min\{d_{ij}^{(1)},\ d_{i2}^{(1)}+d_{2j}^{(1)}\}$$

则$d_{ij}^{(2)}$表示从v_i直接到v_j，或者从v_i经过v_1或v_2或v_1与v_2而到v_j的所有这些路径中最短的一条路径。

用归纳法，设v_i与v_j之间已插入$(k-1)$个结点，则插入第k个结点时得矩阵$D^{(k)}=(d_{ij}^{(k)})_{n\times n}$，其中

$$d_{ij}^{(k)}=\min\{d_{ij}^{(k-1)},\ d_{ik}^{(k-1)}+d_{kj}^{(k-1)}\}\tag{7.3}$$

则$d_{ij}^{(k)}$是从v_i直接到v_j，或者经过v_1，v_2，…v_k中部分或全部结点而到v_j的所有路径中最短的一条路径。

当$k=n$，则$D^{(n)}=(d_{ij}^{(n)})_{n\times n}$即为图的距离矩阵。在求距离矩阵的迭代过程中，也可将路径矩阵P求出来。

弗洛伊得算法：

```
begin
1. for i=1 to n do
2. for j=1 to n do
3. d(i, j)←—a(i, j)
4. if i≠j and a(i, j)<max
     then P (i, j) ←—(i)+(j)
5. for k=1 to n do
6. for i=1 to n do
7. for j=1 to n do
8. if d(i, k)+d (k, j) <d(i, j) then
```

$$d(i,\ j) \longleftarrow d(i,\ k)+d(k,\ j)$$
$$P(i,\ j) \longleftarrow P(i,\ k)+P(k,\ j)$$

end

在上述算法中，$P=(P_{ij})_{n\times n}$为最短路径矩阵。如果不存在从v_i到v_j的路径，则对应的P_{ij}为空。从算法可知，必须计算n个矩阵$D^{(1)}$，$D^{(2)}$，…，$D^{(n)}$，其中每个矩阵包含n^2个元素，因此总共需要计算n^3个元素，而每一次计算都要作一次加法和求一次极小值，因此算法的时间复杂性是$O(2n^3)$级的。但是这一算法有一极大的优点，即只要图不存在负长度的回路，则弧的权值可以允许是负值。

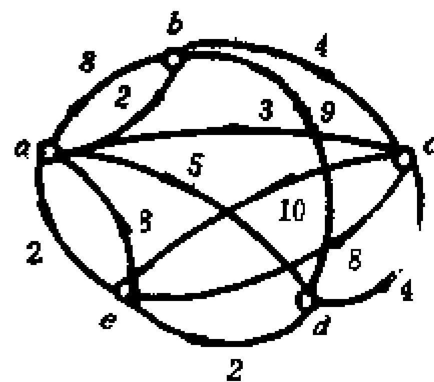


图 7.6

例 7.5 试用弗洛伊得算法求图 7.6 任意两点的距离及最短路径。

解：根据所给图得距离和路径矩阵的初始值如下：

$$D^{(0)}=A=\begin{array}{c} \\ a\\ b\\ c\\ d\\ e\end{array}\begin{array}{c}\begin{array}{ccccc}a & b & c & d & e\end{array}\\ \begin{bmatrix}0 & 8 & 3 & 5 & 2\\ 2 & 0 & \infty & 9 & \infty\\ \infty & 4 & 0 & \infty & 10\\ \infty & \infty & 4 & 0 & \infty\\ 8 & \infty & 8 & 2 & 0\end{bmatrix}\end{array}\qquad P^{(0)}=\begin{array}{c} \\ a\\ b\\ c\\ d\\ e\end{array}\begin{array}{c}\begin{array}{ccccc}a & b & c & d & e\end{array}\\ \begin{bmatrix} & ab & ac & ad & ae\\ ba & & & bd & \\ & cb & & & ce\\ & & dc & & \\ ea & & ec & ed & \end{bmatrix}\end{array}$$

将结点a插入，$D^{(1)}$的各元素和相应的最短路径计算如下（结点a，b，c，d，e按1，2，3，4，5排序）

$$d_{11}^{(1)}=d_{11}^{(0)}=0$$
$$d_{12}^{(1)}=d_{12}^{(0)}=8,\ P_{12}^{(1)}=ab$$
$$d_{13}^{(1)}=d_{13}^{(0)}=3,\ P_{13}^{(1)}=ac$$
$$d_{14}^{(1)}=d_{14}^{(0)}=5,\ P_{14}^{(1)}=ad$$
$$d_{15}^{(1)}=d_{15}^{(0)}=2,\ P_{15}^{(1)}=ae$$
$$d_{21}^{(1)}=d_{21}^{(0)}=2,\ P_{21}^{(1)}=ba$$
$$d_{22}^{(1)}=d_{22}^{(0)}=0$$
$$d_{23}^{(1)}=\min\{d_{23}^{(0)},\ d_{21}^{(0)}+d_{13}^{(0)}\}=\min\{\infty,\ 2+3\}=5$$
$$P_{23}^{(1)}=ba+ac=bac$$
$$d_{24}^{(1)}=\min\{d_{24}^{(0)},\ d_{21}^{(0)}+d_{14}^{(0)}\}=\min\{9,\ 2+5\}=7$$
$$P_{24}^{(1)}=ba+ad=bad$$
$$d_{25}^{(1)}=\min\{d_{25}^{(0)},\ d_{21}^{(0)}+d_{15}^{(0)}\}=\min\{\infty,\ 2+2\}=4$$
$$P_{25}^{(1)}=ba+ae=bae$$
$$d_{31}^{(1)}=d_{31}^{(0)}=\infty$$
$$d_{32}^{(1)}=d_{32}^{(0)}=4,\ P_{32}^{(1)}=cb$$
$$d_{33}^{(1)}=d_{33}^{(0)}=0$$
$$d_{34}^{(1)}=d_{34}^{(0)}=\infty$$
$$d_{35}^{(1)}=d_{35}^{(0)}=10,\ P_{35}^{(1)}=ce$$

$$d_{41}^{(1)}=d_{41}^{(0)}=\infty$$

$$d_{42}^{(1)}=d_{42}^{(0)}=\infty$$

$$d_{43}^{(1)}=d_{43}^{(0)}=4,\quad P_{43}^{(1)}=dc$$

$$d_{44}^{(1)}=d_{44}^{(0)}=0$$

$$d_{45}^{(1)}=d_{45}^{(0)}=\infty$$

$$d_{51}^{(1)}=d_{51}^{(0)}=8,\quad P_{51}^{(1)}=ea$$

$$d_{52}^{(1)}=\min\{d_{52}^{(0)},\ d_{51}^{(0)}+d_{12}^{(0)}\}=\min\{\infty,\ 8+8\}=16$$

$$P_{52}^{(1)}=ea+ab=eab$$

$$d_{53}^{(1)}=\min\{d_{53}^{(0)},\ d_{51}^{(0)}+d_{13}^{(0)}\}=\min\{8,\ 8+3\}=8$$

$$=d_{53}^{(0)},\quad P_{53}^{(1)}=P_{53}^{(0)}=ec$$

$$d_{54}^{(1)}=d_{54}^{(0)}=2,\quad P_{54}^{(1)}=ed$$

$$d_{55}^{(1)}=d_{55}^{(0)}=0$$

由此得到$D^{(1)}$和$P^{(1)}$如下：

$$D^{(1)}=\begin{bmatrix}0 & 8 & 3 & 5 & 2\\ 2 & 0 & 5 & 7 & 4\\ \infty & 4 & 0 & \infty & 10\\ \infty & \infty & 4 & 0 & \infty\\ 8 & 16 & 8 & 2 & 0\end{bmatrix}\quad P^{(1)}=\begin{bmatrix} & ab & ac & ad & ae\\ ba & & bac & bad & bae\\ & cb & & & ce\\ & & dc & & \\ ea & eab & ec & ed & \end{bmatrix}$$

在 $D^{(1)}$上将第二个结点 b 插入，显然有

$$d_{ii}^{(2)}=d_{ii}^{(1)},\quad i=1,\ 2,\ \cdots 5$$

$$d_{2j}^{(2)}=d_{2j}^{(1)},\quad j=1,\ 2,\ \cdots,\ 5$$

$$d_{i2}^{(2)}=d_{i2}^{(1)},\quad i=1,\ 2,\ \cdots,\ 5$$

因此我们只须计算其余元素如下：

$$d_{13}^{(2)}=d_{13}^{(1)}=3,\quad P_{13}^{(2)}=ac$$

$$d_{14}^{(2)}=d_{14}^{(1)}=5,\quad P_{14}^{(2)}=ad$$

$$d_{15}^{(2)}=d_{15}^{(1)}=2,\quad P_{15}^{(2)}=ae$$

$$d_{31}^{(2)}=\min\{d_{31}^{(1)},\ d_{32}^{(1)}+d_{21}^{(1)}\}=\min\{\infty,\ 4+2\}=6$$

$$P_{31}^{(2)}=P_{32}^{(1)}+P_{21}^{(1)}=cb+ba=cba$$

$$d_{34}^{(2)}=\min\{d_{34}^{(1)},\ d_{32}^{(1)}+d_{24}^{(1)}\}=\min\{\infty,\ 4+7\}=11$$

$$P_{34}^{(2)}=P_{32}^{(1)}+P_{24}^{(1)}=cb+bad=cbad$$

$$d_{35}^{(2)}=\min\{d_{35}^{(1)},\ d_{32}^{(1)}+d_{25}^{(1)}\}=\min\{10,\ 4+4\}=8$$

$$P_{35}^{(2)}=P_{32}^{(1)}+P_{25}^{(1)}=cb+bae=cbae$$

$$d_{41}^{(2)}=d_{41}^{(1)}=\infty$$

$$d_{43}^{(2)}=d_{43}^{(1)}=4,\quad P_{43}^{(2)}=dc$$

$$d_{45}^{(2)}=d_{45}^{(1)}=\infty$$

$$d_{51}^{(2)}=d_{51}^{(1)}=8,\quad P_{51}^{(2)}=ea$$

$$d_{53}^{(2)}=d_{53}^{(1)}=8,\quad P_{53}^{(2)}=ec$$

$$d_{54}^{(2)}=d_{54}^{(1)}=2,\quad P_{54}^{(1)}=ed$$

于是得到$D^{(2)}$和$P^{(2)}$如下：

$$D^{(2)}=\begin{bmatrix} 0 & 8 & 3 & 5 & 2 \\ 2 & 0 & 5 & 7 & 4 \\ 6 & 4 & 0 & 11 & 8 \\ \infty & \infty & 4 & 0 & \infty \\ 8 & 16 & 8 & 2 & 0 \end{bmatrix} \quad P^{(2)}=\begin{bmatrix} & ab & ac & ad & ae \\ ba & & bac & bad & bae \\ cba & cb & & cbad & cbae \\ & & dc & & \\ ea & eab & ec & ed & \end{bmatrix}$$

同理可求出$D^{(3)}$、$P^{(3)}$，$D^{(4)}$、$P^{(4)}$，$D^{(5)}$、$P^{(5)}$如下，最后得到的$D^{(5)}$和$P^{(5)}$即为距离矩阵和最短路径矩阵。

$$D^{(3)}=\begin{bmatrix} 0 & 7 & 3 & 5 & 2 \\ 2 & 0 & 5 & 7 & 4 \\ 6 & 4 & 0 & 11 & 8 \\ 10 & 8 & 4 & 0 & 12 \\ 8 & 12 & 8 & 2 & 0 \end{bmatrix} \quad P^{(3)}=\begin{bmatrix} & acb & ac & ad & ae \\ ba & & bac & bad & bae \\ cba & cb & & cbad & cbae \\ dcba & dcb & dc & & dcbae \\ ea & ecb & ec & ed & \end{bmatrix}$$

$$D^{(4)}=\begin{bmatrix} 0 & 7 & 3 & 5 & 2 \\ 2 & 0 & 5 & 7 & 4 \\ 6 & 4 & 0 & 11 & 8 \\ 10 & 8 & 4 & 0 & 12 \\ 8 & 10 & 6 & 2 & 0 \end{bmatrix} \quad P^{(4)}=\begin{bmatrix} & acb & ac & ad & ae \\ ba & & bac & bad & bae \\ cba & cb & & cbad & cbae \\ dcba & dcb & dc & & dcbae \\ ea & edcb & edc & ed & \end{bmatrix}$$

$$D^{(5)}=\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array}\begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 7 & 3 & 4 & 2 \\ 2 & 0 & 5 & 6 & 4 \\ 6 & 4 & 0 & 10 & 8 \\ 10 & 8 & 4 & 0 & 12 \\ 8 & 10 & 6 & 2 & 0 \end{bmatrix} \end{array} \quad P^{(5)}=\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array}\begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} & acb & ac & aed & ae \\ ba & & bac & baed & bae \\ cba & cb & & cbaed & cbae \\ dcba & dcb & dc & & dcbae \\ ea & edcb & edc & ed & \end{bmatrix} \end{array}$$

§7.3 最 优 路 径

在研究各种工程、经济等决策问题时，常常用图作为一个数学模型进行描述，这种图称为决策图，它不同于一般的有向图，有其自身的特点。

定义 7.5 决策图$D=(V,\ E)$是一个不含回路的带权有向图，并且

（1）它的结点集合V分成S个子集V_1，V_2，…，V_s，并有

$$V_1 \cup V_2 \cup \cdots \cup V_s = V$$

$$V_i \cap V_j = \phi,\ i \neq j,\ 1 \leqslant i,\ j \leqslant S$$

（2）每一结点子集V_i称为一级，V_1为第1级，只有一个结点，称为初始结点，V_s为最后一级，可以有不止一个结点，均称为得胜结点。

（3）对任一条有向边$\langle v_i,\ v_j\rangle \in E$，如$v_i \in V_i$，则必有$v_j \in V_{i+1}$

例如图7.7表示一个决策图，它有5级，由于决策图常用作描述对策，所以常采用对策中的一些名词术语，即将结点称为布局，初始结点称为初始布局，得胜结点称为得胜布局，一条弧称为一个棋步，弧上所带的权称为这个棋步的成本。

定义 7.6 在决策图$D=(V,E)$中，如果存在一条从初始结点到得胜结点的路径，则

称图是可解的，并称这样的路径为可解路径。

例如图7.7的决策图是可解的，其中$v_{11}\to v_{21}\to v_{32}\to v_{43}\to v_{53}$是一条可解路径。一般情况下，可解路径不止一条。

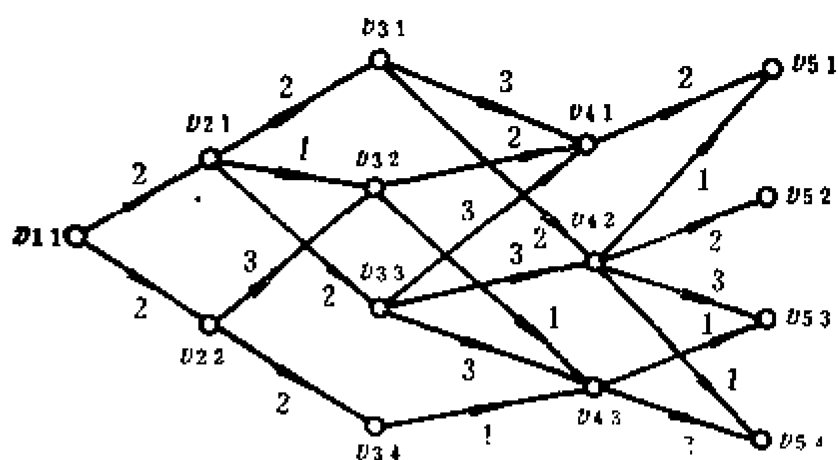


图 7.7

一条路径上各边成本之和，称为这条路径的总成本。

定义 7.7 在决策图$D=(V,E)$的所有可解路径中，总成本最小的可解路径，称为决策图的最优路径。

寻求决策图的最优路径，它的意义是不言而喻的。可以说我们研究一个决策图，目的就是要找到一条最优路径，但是如果企图把所有的可解路径都找出来，然后从中选出总成本最小的，由此得到最优路径，这种方法显然是不可取的。这一节就是探讨如何使用更为简捷有效的方法，求出决策图的最优路径，为此先介绍最优原理。

定义 7.8 在决策图$D=(V,E)$中，从结点v_{ij}到某一得胜结点v_{sk}的路径P，如果是从v_{ij}到V_s的所有得胜点的路径中成本最小的，则称P为结点v_{ij}的最优路径。

定理 7.1 （最优化原理）设路径

$$P=v_{ij},\ v_{(i+1)k},\ v_{(i+2)m},\ \cdots,\ v_{(s-1)r},\ v_{st}$$

是结点v_{ij}的一条最优路径，则路径

$$P'=v_{(i+1)k},\ v_{(i+2)m},\ \cdots,\ v_{(s-1)r},\ v_{st}$$

必然是结点$v_{(i+1)k}$的最优路径。

证： 用反证法，设P'不是结点$v_{(i+1)k}$的最优路径，则在图中必然存在另一条路径是它的最优路径，设这条路径为

$$\hat{P}=v_{(i+1)k},\ v_{(i+2)m'},\cdots,v_{(s-1)r'},\ v_{st'}$$

记路径P的总成本为$K(P)$，路径P'的总成本为$K(P')$，路径$\hat{P}$的总成本为$K(\hat{P})$，则应有

$$K(\hat{P})<K(P') \qquad (A)$$

设弧$(v_{ij},\ v_{(i+1)k})$的权为$w(v_{ij},\ v_{(i+1)k})$，则

$$K(P)=K(P')+w(v_{ij},\ v_{(i+1)k}) \qquad (B)$$

由于结点$v_{(i+1)k}$存在另一条路径$\hat{P}$，则v_{ij}也存在另一条路径$\hat{P}'$

$$\hat{P}'=v_{ij},\ v_{(i+1)k},\ v_{(i+2)m'},\ \cdots,\ v_{(s-1)r'},\ v_{st'}$$

设它的成本为$K\hat{P}')$，则

$$K(\hat{P}')=K(\hat{P})+w(v_{ij},v_{(i+1)k}) \qquad (C)$$

比较(A),(B),(C)三式即得

$$K(\hat{P}')<K(P)$$

与P是v_{ij}的最优路径矛盾。所以P'是结点$v_{(i+1)k}$的最优路径。 ▌

这一定理启示我们，寻找结点v_{ij}的最优路径应该先求出它所能到达的下一级结点的最优路径。

如图7.8，结点v_{ij}有m条弧引向下一级的结点，每一条弧上的成本分别记为$K_1,K_2,\cdots,$

K_m，弧的终端结点为$v_{(i+1)1}, v_{(i+1)2}$，…，$v_{(i+1)m}$，它们的最优路径成本分别记作$K(1)$，$K(2)$，…，$K(m)$，则v_{ij}的最优路径一定是这m条路径中的一条，它可以从成本最小的路径中求得，即如果P是v_{ij}的最优路径，则必有

$$K(P)=\min\{K_1+K(1),\ K_2+K(2),\ \cdots, K_m+K(m)\} \qquad (7.4)$$

即成本最小的那条路径就是P

图 7.8

定义 7.9 在决策图$D=(V,E)$中结点v_{ij}的最优路径的总成本，称为结点v_{ij}的成本，记作K_{ij}。

至此，我们得出求决策图最优路径的算法如下：

1．令得胜级的每一结点的成本为零，即

$$K_{sl}=0,\ 1\leqslant l\leqslant m,\ m\text{为得胜级的结点数}$$

2．$i \longleftarrow s-1$

3．在V_i级中按公式（7.4）计算各结点的成本及最优路径。

4．$i \longleftarrow i-1$

5．如$i=0$，停止，否则转步骤（3）

例 7.6 求出图7.9的决策图的最优路径。

解：

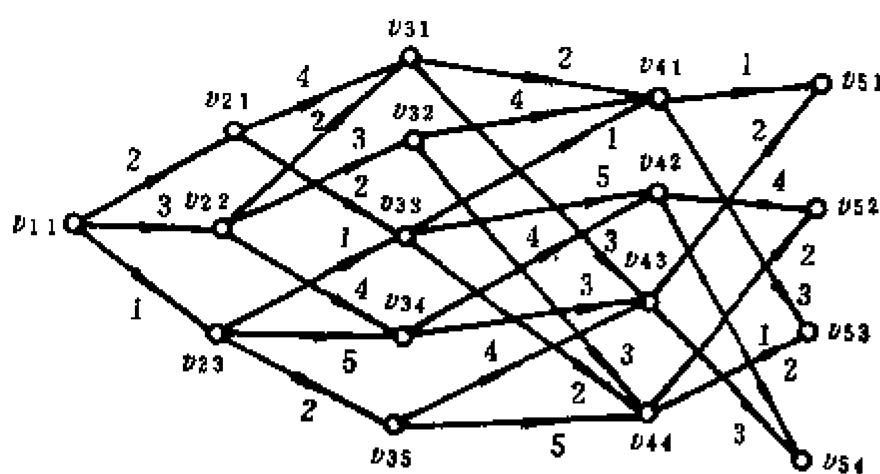
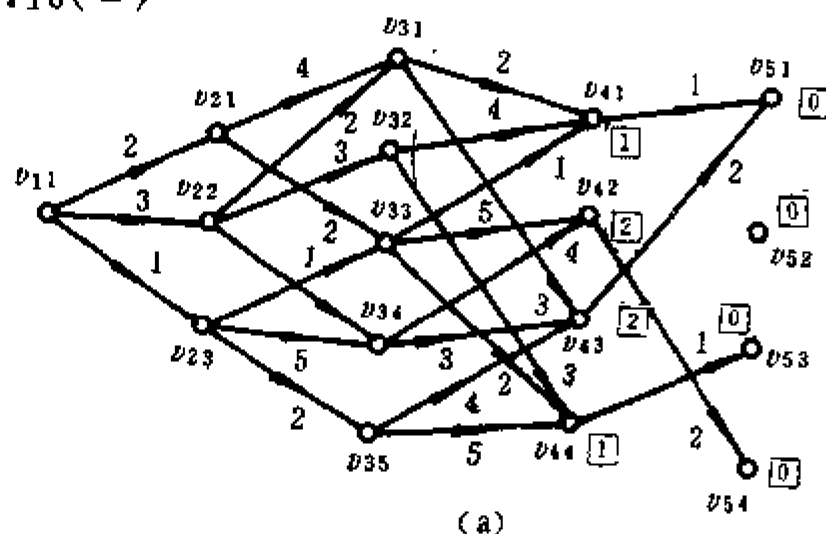


图 7.9

1．各得胜结点的成本都标0，并由此得出第4级各结点的成本及最优路径（非最优路径不画出）得图7.10（a）



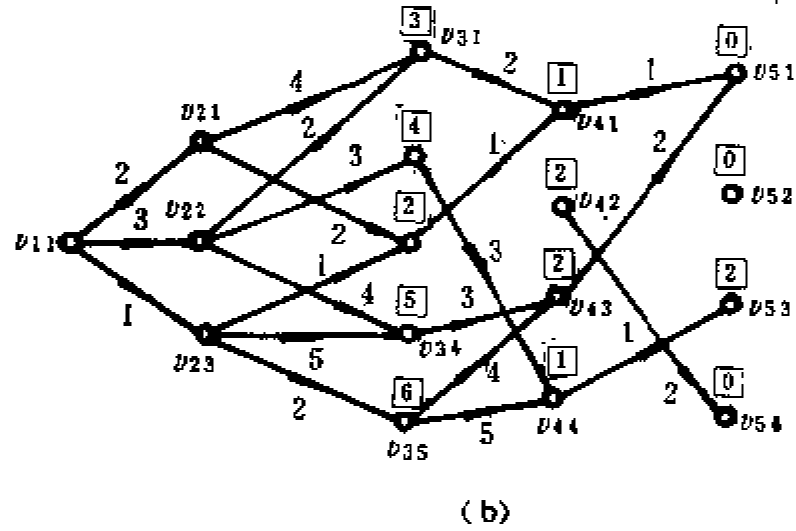
（a）

图 7.10

2．由式（7.4）计算V_3级各结点成本如下

$$K_{31}=\min\{2+1,3+2\}=3$$
$$K_{32}=\min\{4+1,3+1\}=4$$
$$K_{33}=\min\{1+1,5+2,2+1\}=2$$
$$K_{34}=\min\{4+2,3+2\}=5$$
$$K_{35}=\min\{4+2,5+1\}=6$$

将各结点成本标在图上，保留最优路径得到图（b）

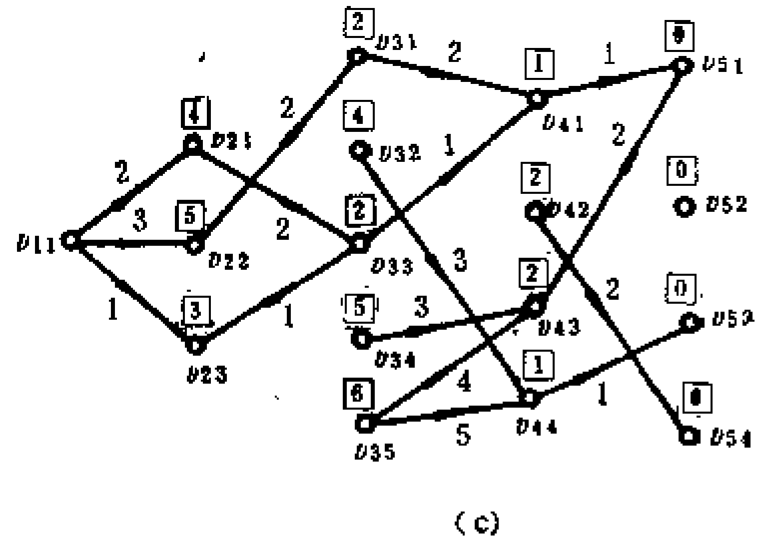


（b）

3．V_2级各结点成本计算如下：

$$K_{21}=\min\{4+3,2+2\}=4$$
$$K_{22}=\min\{2+3,3+4,4+5\}=5$$
$$K_{23}=\min\{1+2,5+5,2+6\}=3$$

将各结点的成本标在图上，保留最优路径，得到图（c）

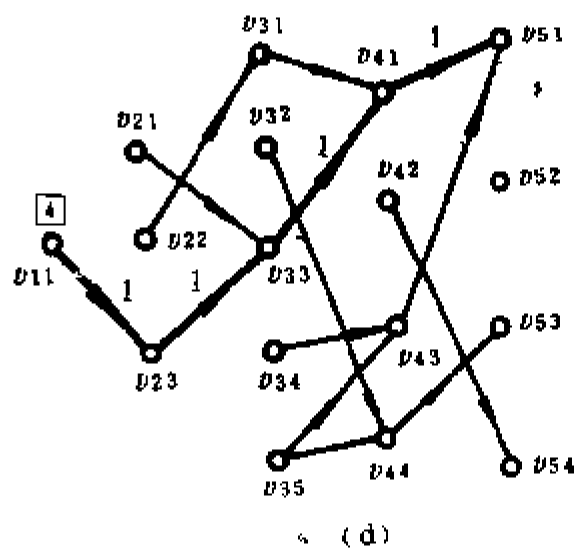


（c）

4．计算初始级结点成本

$$K_{11}\min\{2+4,3+5,1+3\}=4$$

得到图（d）



（d）

图 7.10

在保留结点的最优路径情况下，初始结点的可解路径（可能不止一条）就是图的最优路径。如图(d)，决策图的最优路径为

$$P=v_{11},v_{23},v_{33},v_{41},v_{51}$$

其总成本为4

§7.4 关键路径

一、工程的统筹规划

图论在运筹学中的应用是很广泛的，其中之一就是对一项巨大而复杂的工程项目的计划安排，统筹规划，常称为统筹法或关键路径法，用以描述的图也叫评审图（PERT图）或统筹网络。

统筹网络是由一个无回路、无平行边和自环的有限带权有向图构成。如图7.11所示。

图中仅有一个引入次数为0的结点 v_1，称为始点，仅有一个引出次数为0的结点 v_6，称为收点，图的每一条弧表示一项活动（或程序、工序、任务等）它所带的权表示完成这项活动须要的时间，如权值为0则表示完成该项活动所须要的时间可以忽略不计，每一结点称为一个事件，

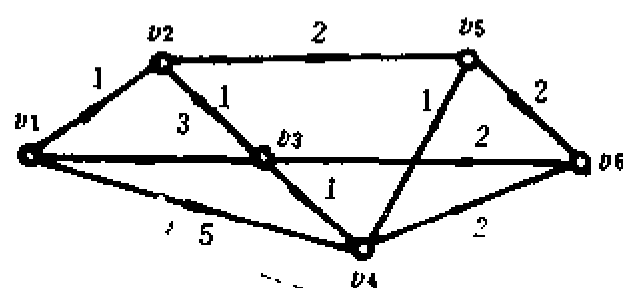


图 7.11

表示时间的瞬态。终止于这一结点的弧表示对应于这条弧所表示的活动到此瞬间结束，起始于这一结点的弧表示对应这条弧的活动从此瞬间开始，始点表示整个工程的开始，收点表示整个工程的完成。

弧的邻接关系表示各项活动的依赖关系，即必须终止于结点的所有活动（结点的引入弧）都完成以后，由该结点引出的活动（结点的引出弧）方能开始。例如图中工序 $\langle v_1, v_2\rangle$ 完成之后，工序 $\langle v_2, v_3\rangle$ 和 $\langle v_2, v_5\rangle$ 即可开始，而工序 $\langle v_3, v_4\rangle$ 能够开始之前，工序 $\langle v_1, v_3\rangle$ 和 $\langle v_2, v_3\rangle$ 必须都要完成。

对于统筹网络，主要研究的问题为：

（1）完成整个工程至少需要的时间

（2）哪些工序（活动、任务）是影响整个工程能否按期完成的关键

二、最早完成时间

定义 7.10 设P是始点v_1到事件 v_j 的任意一条路径，$t_p(v_j)$是从v_1沿着路径P到达v_j所需的时间，则称 $t_p(v_j)$的最大值为事件 v_j 的最早完成时间，记作$TE(v_j)$，即

$$TE(v_j)=\max\{t_p(v_j)\},\ j\neq 1$$

以图7.11为例，工序从始点v_1开始，同时进行三道工序，即 $\langle v_1, v_2\rangle$，$\langle v_1, v_3\rangle$ 与 $\langle v_1, v_4\rangle$，完成工序 $\langle v_1, v_2\rangle$ 需要1个单位时间，比方说1个月，由于 v_1 到 v_2 只有一条路径，因而 v_2 的最早完成时间为1个月。这一事件结束，接着下面两道工序 $\langle v_2, v_3\rangle$ 和 $\langle v_2, v_5\rangle$ 开始，工序 $\langle v_2, v_3\rangle$ 的完成时间是1个月，所以从工程开工（v_1）算起到完成工序 $\langle v_2, v_3\rangle$，需

要的时间是1+1=2个月，而工序(v_1,v_3)的完成时间是3个月，只有工序(v_2,v_3)和(v_1,v_3)都完成了，事件v_3才算结束，所以事件v_3的最早完成时间必须是3个月，不能比它少，否则工序(v_1,v_3)还未完成，下面的工序不能开始，这就是事件最早完成时间的涵意，也是在v_1到v_j的诸多路径P中取$t_p(v_j)$最大值的原因。

由此可知，一个事件的最早完成时间，是指完成这一事件最少需要的时间，不能比它再少，否则这一事件将不能完成。

显然，始点v_1的最早完成时间为0，即$TE(v_1)=0$，而收点v_s的最早完成时间$TE(v_s)$即是整个工程的最早完成时间，或工程的计划完成时间，求出这个时间，我们对工程的统筹规划就能做到心里有数。

怎样计算工程的最早完成时间呢？从上面的分析我们可以看到，求一个事件v_i的最早完成时间应从前面开始。

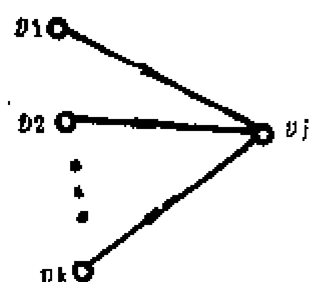


图 7.12

如图7.12，设终止于v_j的弧共有k条，它们的始点分别记为$v_1,v_2\cdots v_k$，这些点的最早完成时间分别记为$TE(v_1)$，$TE(v_2),\cdots,TE(v_k)$，每条弧所带的权记作$w(v_i,v_j)$，$w(v_2,v_j)$，$\cdots,w(v_k,v_j)$，则v_j的最早完成时间可按下式计算

$$TE(v_j)=\max_{(v_i,v_j)\in E}\{TE(v_i)+w(v_i,v_j)\} \tag{7.5}$$

这个公式告诉我们，计算事件的最早完成时间应从始点开始，顺着弧的方向顺序求出。

例 7.7 求出图7.11中各结点的最早完成时间。

解： 从始点开始，有

$TE(v_1)=0$

$TE(v_2)=\max\{TE(v_1)+w(v_1,v_2)\}=\max\{0+1\}=1$

$TE(v_3)=\max\{TE(v_1)+w(v_1,v_3),\ TE(v_2)+w(v_2,v_3)\}$

$\quad=\max\{0+3,1+1\}=3$

$TE(v_4)=\max\{TE(v_1)+w(v_1,v_4),\ TE(v_3)+w(v_3,v_4)\}$

$\quad=\max\{0+5,3+1\}=5$

$TE(v_s)=\max\{TE(v_2)+w(v_2,v_5),\ TE(v_4)+w(v_4,v_5)\}$

$\quad\max\{1+2,5+1\}=6$

$TE(v_6)=\max\{TE(v_3)+w(v_3,v_6),\ TE(v_4)+w(v_4,v_6),\ TE)v_5)$

$\quad+w(v_5,v_6)\}$

$\quad=\max\{3+2,5+2,6+2\}=8$

三、最晚完成时间

定义 7.11 在收点v_s的最早完成时间不增加的前提下，允许从始点v_1最晚到达v_j的时间，称为v_j的最晚完成时间，记作$TL(v_j)$

从定义可知对始点v_1和收点v_s，都有

$$TL(v_1)=TE(v_1)=0$$
$$TL(v_s)=TE(v_s)$$

设 P' 是从 v_j 到收点 v_s 的任意一条路径，$t_{P'}(v_j)$ 是从 v_j 沿路径 P' 到达收点 v_s 的时间，为了保证整个工程的计划完成时间不受拖延，确定 v_j 的最晚完成时间必须考虑从 v_j 到收点 v_s 的各条路径（即各道工序）完成时间的最大值。仍以图 7.11 为例，我们考查结点 v_4，它到收点 v_6 有两条路径，一条是 $v_4 \to v_5 \to v_6$，完成这条路径上的各道工序总的时间需要 3 个月。另一条路径是 $v_4 \to v_6$，完成这条路径上的工序需要 2 个月，而整个工程计划完成时间是 8 个月，如果只考虑后一条路径，v_4 的完成时间可以是 6 个月，这样加上工的 $\langle v_4, v_6 \rangle$ 的时间 2 个月，总计 8 个月，没有超过工程的计划完成时间。但是从前一条路径来考虑，显然就不行了，如果 v_4 6 个月完成，再加上工序 $\langle v_4, v_5 \rangle$ 和 $\langle v_5, v_6 \rangle$ 共需 3 个月，则工程要 9 个月才能完成，超过了工程的计划完成时间，可见为了保证工程如期完成，v_4 的完成时间最迟不得超过 $8-3=5$ 个月，这就是结点最晚完成时间的涵意。

由此可知，结点 v_j 的最晚完成时间，是由 v_j 到收点 v_s 的各条路径中最大活动时间所决定的，它们有如下关系

$$TL(v_j) = TE(v_s) - \max\{t_{P'}(v_j)\}$$

直接应用上式计算各结点的最晚完成时间是很麻烦的，但是上式给了我们一个启示，即计算结点的最晚完成时间，应从收点开始，由后往前地逐点计算。

如图7.13，设从结点 v_j 引出的弧共有 k 条，即 $\langle v_j, v_1 \rangle$，$\langle v_j, v_2 \rangle$，…，$\langle v_j, v_k \rangle$，它们分别终止于结点 $v_1, v_2, \cdots, v_k$ 上，用 $w(v_j, v_1)$，$w(v_j, v_2)$，…，$w(v_j, v_k)$ 分别表示各条弧上的权，各终止结点上的最晚完成时间记作 $TL(v_i)$ 则 v_j 的最晚完成时间 $TL(v_j)$ 可按下式求出

$$TL(v_j) = \min_{\langle v_j, v_i \rangle \in E}\{TL(v_i) - w(v_j, v_i)\} \qquad (7.6)$$

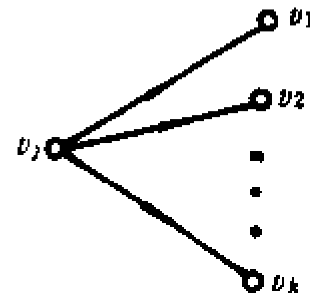


图 7.13

例 7.8 求出图7.11中各结点的最晚完成时间。

解： 从收点 v_6 开始，有

$$TL(v_6) = TE(v_6) = 8$$

$$TL(v_5) = \min\{TL(v_6) - w(v_5, v_6)\} = \min\{8-2\} = 6$$

$$TL(v_4) = \min\{TL(v_6) - w(v_4, v_6),\ TL(v_5) - w(v_4, v_5)\} = \min\{8-2, 6-1\} = 5$$

$$TL(v_3) = \min\{TL(v_6) - w(v_3, v_6), TL(v_4) - w(v_3, v_4)\} = \min\{8-2, 5-1\} = 4$$

$$TL(v_2) = \min\{TL(v_5) - w(v_2, v_5),\ TL(v_3) - w(v_2, v_3)\} = \min\{6-2, 4-1\} = 3$$

$$TL(v_1) = \min\{TL(v_4) - w(v_1, v_4),\ TL(v_3) - w(v_1, v_3),\ TL(v_2) - w(v_1, v_2)$$

$$\min\{5-5, 4-3, 3-1\} = 0 = TE(v_1)$$

四、缓冲时间和关键路径

定义 7.12 事件 v_j 的最晚完成时间与最早完成时间之差，称为 v_j 的缓冲时间，记作 $t(v_j)$，即

$$t(v_j)=TL(v_j)-TE(v_j) \tag{7.7}$$

v_j的缓冲时间，表示事件v_j可以拖延时间$t(v_j)$完成而不致影响整个工程的计划完成时间。

如图7.11，各结点的缓冲时间为：

$t(v_1)=0$

$t(v_2)=TL(v_2)-TE(v_2)=3-1=2$

$t(v_3)=TL(v_3)-TE(v_3)=4-3=1$

$t(v_4)=TL(v_4)-TE(v_4)=5-5=0$

$t(v_5)=TL(v_5)-TE(v_5)=6-6=0$

$t(v_6)=TL(v_6)-TE(v_6)=0$

定义 7.13 从始点v_1到收点v_t的路径P_t，如果满足：

（1）P_t上的所有结点v_i，都有$TL(v_i)=TE(v_i)$

（2）P_t上的所有弧$\langle v_i,v_j\rangle$，都有$w(v_i,v_j)=TE(v_j)-TE(v_i)$

则称P_t为评审图的关键路径。

我们把结点v_j的最早完成时间和最晚完成时间以$[TE(v_j),TL(v_j)]$的形式标在结点v上，图7.11可以画成图7.14

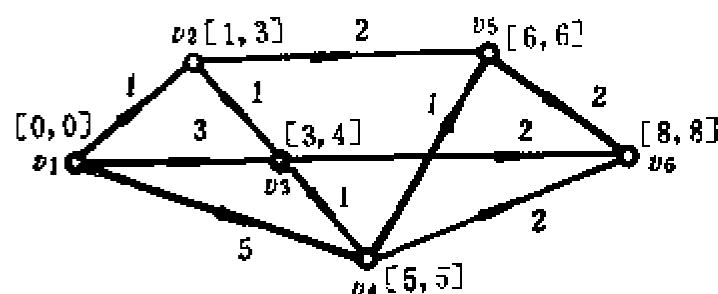


图 7.14

从图中即可得到关键路径为（图中粗线所示）

$$P_t=v_1\to v_4\to v_5\to v_6$$

由关键路径的定义可知：

（1）关键路径上各道工序完成时间的总和，即为整个工程的计划完成时间。

（2）关键路径上的任何一道工序，如果延误了完成时间，整个工程的计划完成时间也将推迟。反之，如果这些工序能提前完成，则工程的计划完成时间也可提前。所以关键路径上的各道工序对工程的按期完成与否至关重要，应作为工程的重点项目给予充分注意和保证。

（3）如果把各条弧的权看作长度，则关键路径是图中长度最长的一条路径。

（4）在复杂的工程中，评审图的关键路径可以不止一条。

习题与思考题

1．求图7.15中结点a至其余结点的最短路径及距离。

2．求图7.16中结点a至其余各点的最短路径及距离。

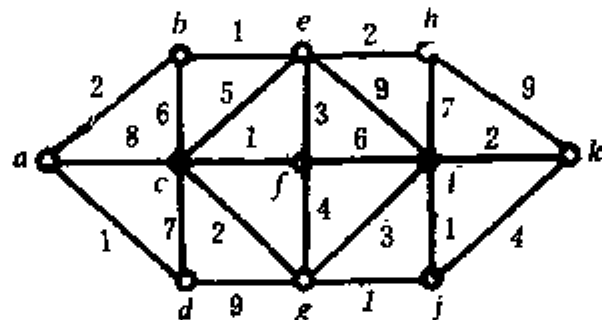


图 7.15

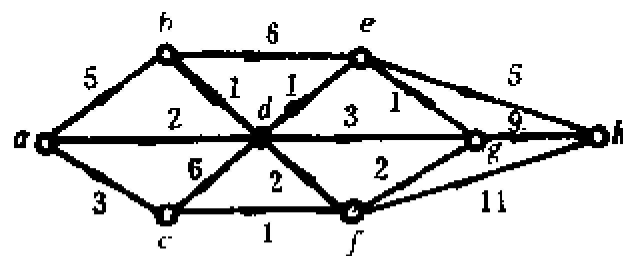


图 7.16

3．某公司在六个城市 $C_1, C_2, \cdots, C_6$ 中都有分公司，从C_i到C_j的直接航班票价由下述矩阵的第(i,j)元素给出（∞表示无直接航班），该公司想算出一张任意两个城市之间的最廉航价路线表，试作出这样的表来。

$$\begin{bmatrix} 0 & 50 & \infty & 40 & 25 & 10 \\ 50 & 0 & 15 & 20 & \infty & 25 \\ \infty & 15 & 0 & 10 & 20 & \infty \\ 40 & 20 & 10 & 0 & 10 & 25 \\ 25 & \infty & 20 & 10 & 0 & 55 \\ 10 & 25 & \infty & 25 & 55 & 0 \end{bmatrix}$$

4．试举例说明当带权有向图如果有某些弧的权为负值时，应用得克斯特拉最短路算法是否合适。

5．当带权有向图存在弧权为负值时，试设计一个求最短路的算法。

6．对各结点任意编号是否影响弗洛伊得算法的效率？

7．试用矩阵幂的算法，求图7.17中任意两结点的距离及最短路径。

8．试用弗洛伊得算法求图7.17中任意两点之间的距离及最短路径。

9．炼油厂各个油罐之间的运费率如下

油罐	A	B	C	D
A	0.00	0.13	0.14	0.15
B	0.08	0.00	0.13	0.08
C	0.17	0.12	0.00	0.18
D	0.10	0.06	0.13	0.00

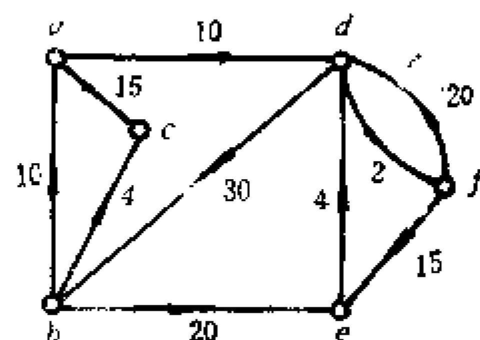


图 7.17

从油罐C将油运到油罐D，求最佳运输路线。如果这条路线遭到破坏，试求一次佳的运输路线。

10．某旅店经理须安排好下个月结婚房间的预订。他已收到许多预订要求，到达和离去日期各不相同。旅店对各种预订可得到不同的收入，这是因为对学生、雇员、航空公司人员等等规定了不同的收费。怎样用得克斯特拉算法求出结婚房间的最佳预订计划，使旅店可以获得最大利润。

（提示：用连结到达日期和离去日期的一条弧表示每一个预订要求，最终得到的图将不包含回路，在这种情况下，可应用一种经修改的得克斯特拉算法）

11．如果带权有向图某些弧的权值为负但没有负回路，证明弗洛伊得的最短路算法仍然是正确的。如果出现负回路，情况会发生什么变化？

12．图7.18给出了一个决策图，试求其最优路径。

13．图7.19给出了一个评审图，试求出各结点的最早完成时间、最晚完成时间和缓冲时间，完成计划的最短时间及关键路径。

14．五项任务A，B，C，D和E构成某工程，它们之间的关系为：B和D在A完工之后才能开工，E和D在C完工之后才能开工，试绘出符合上述要求的任务网络。

15．某公司已决定引进一套新的、更为有效的计算机系统。这样就不仅有新系统的安装阶段，还将有旧系统的拆除阶段，所有这些都必须完成，在任何时候至少要有一套系统

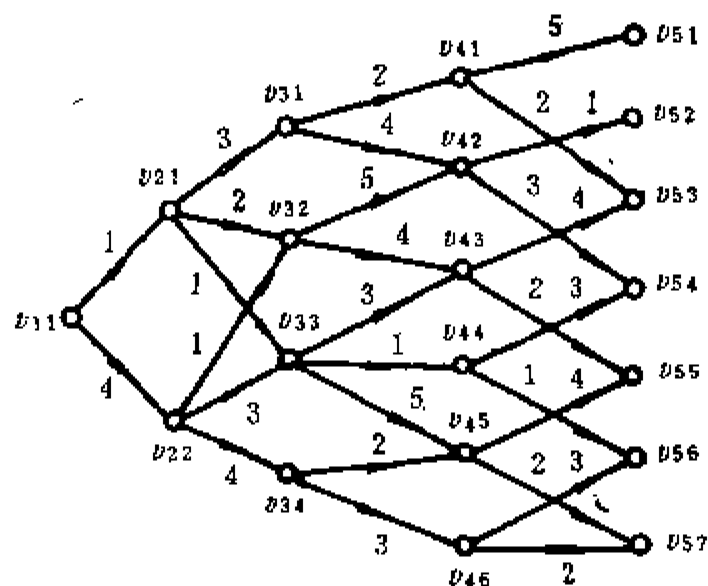


图 7.18

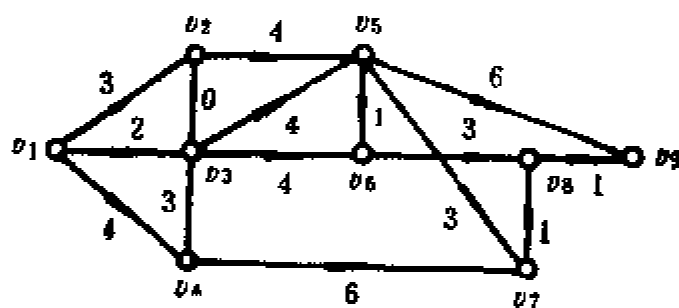


图 7.19

在运转。而且已存入的程序必须转换成新系统的语言，人员必须受到培训以便能操作新系统。

试构造一个详细的统筹网络，以便将这一过程表示出来。

16．对于图7.20所示的统筹网络，如果活动 $\langle v_3, v_4\rangle$ 的时间改为 5，对原计划的各项结果有无影响？这项活动的时间可以增加多少而不致改变整个工程的最早完成时间。

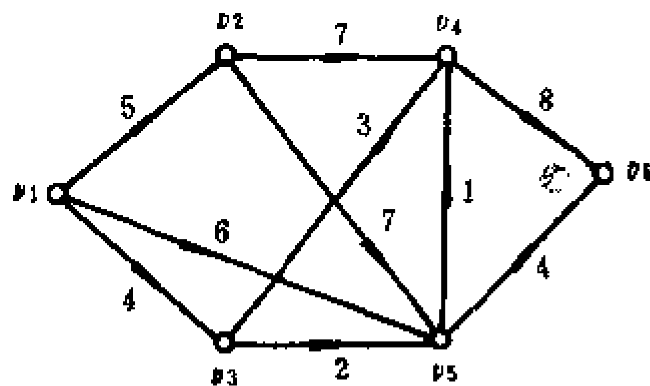


图 7.20

17．设 $w(v_i, v_j)$ 表示活动 $\langle v_i, v_j\rangle$ 的完成时间，$TL(v_i)$ 为 v_i 的最晚完成时间，$TE(v_j)$ 为 v_j 的最早完成时间，称

$$t_{ij} = TE(v_j) - TL(v_i) - w(v_i, v_j)$$

为活动 $\langle v_i, v_j\rangle$ 的独立浮动时间。

试解释 t_{ij} 的涵义并举例说明。

18．试解释：如果活动 $\langle v_i, v_j\rangle$ 的独立浮动时间 t_{ij} 为负值将对工程筹划有何影响。

19．构造一个统筹网络，在网络中有一项活动有具有负值的独立浮动。对这个值加以解释。

第八章 回路问题

在第二章的最末一节中，我们曾讨论了图的遍历。这一章我们将从另一个角度研究图的遍历问题，即不重复地连续遍历图的所有边或结点，由此引出两种特殊形式的图——E图和H图。它不仅是图论最早出现的问题，对图论的发展也起着很大的推动作用。

§8.1 E图和M图

人们常把哥尼斯堡七桥问题作为图论创立的始源。

哥尼斯堡（KÖnisberg），18世纪是东普鲁士的一座城市，在那里有一条名叫普雷格尔（Pregel）河穿过该城，河中有两个小岛，河的两岸和两个岛由七座桥彼此连通，如图8.1(a)所示。

当时人们提出一个问题：是否存在一条闭合路径，能够从某一地点出发，通过每座桥一次且仅一次最后回到原地。

1736年，欧拉（L. Euler）发表了图论的第一篇论文："哥尼斯堡七桥问题"，确定了这个问题是没有解的，即无法实现从某一地点出发，经过每座桥一次且仅一次而能够回到原地。为了证明这个结论，欧拉用A, B, C, D四个结点表示陆地（河两岸及两个小岛）用两点之间的连线表示跨接两地的桥，于是得到一个表示七桥问题的无向图如图8.1(b)所示，七桥问题就成为这样一个图论问题：从任一结点出发，经过每条边一次且仅一次，最后回到原点。欧拉指出要实现上述要求，只有图的每个结点都与偶数条边关联才有可能。上面的图不具备这个条件，因而是不能实现的。

定义 8.1 一个不含孤点的无向图$G=(V, E)$，如果存在经过每条边一次且仅一次的回路，则称此回路为欧拉回路。

如图8.2的(a)图存在欧拉回路，而(b)图则不存在欧拉回路。

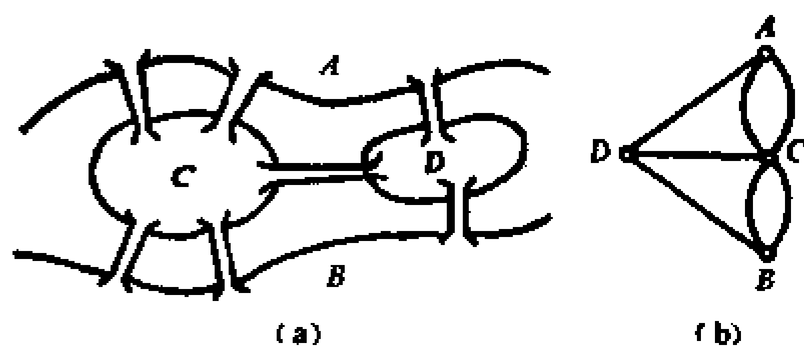


图 8.1

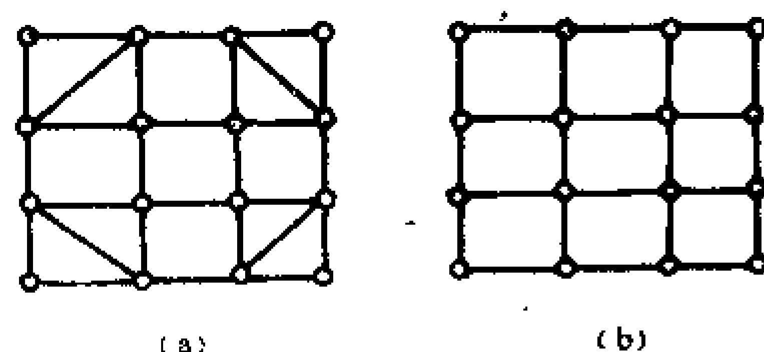


图 8.2

必须把欧拉回路与基本回路或简单回路的概念区别开来。基本回路或简单回路只是对形成回路的边提出要求（边不重复）并未要求必须通过图的所有的边。而欧拉回路对此是要求的，从这一角度来看，欧拉回路是通过图的所有边的简单回路。

判断一个图是否具有欧拉回路，一个简单有效的方法，就是前面提到的欧拉所给出的

论证，即下面的定理。

定理 8.1 一个连通无向图 $G=(V,E)$ 具有欧拉回路的充要条件是它的每一个结点的次数均为偶数。

证：必要性，设 G 具有欧拉回路，用 C 表示这个回路，故 G 的所有结点都在这个回路上，对任一结点 $v\in V$，回路 C 至少穿过它一次，（到达该结点和离开该结点称为穿过结点一次）否则这个结点将是孤点，与题设矛盾，而每穿过结点一次都将使结点的次数增加2，因边是不重复的，故结点的次数一定是2的整数倍，即为偶数。

充分性。设图 G 的每一结点次数均为偶数。则可能有以下两种情况。

（1）图的每一结点次数都是最小的偶数2，由于图是连通的，因而只能有一个回路，即图是一个多边形，显然这个回路就是欧拉回路。

（2）图的某些结点（或者所有结点）的次数是大于2的偶数，不妨用图8.3(a)来描述，则图至少有一个基本回路，例如图中的 *bdeb*，用 C_1 表示这个基本回路，从图 G 中去掉基本回路 C_1，得到子图 G_1，如图(b)所示，则 G_1 各结点的次数仍然都是偶数，如果 G_1 已是一个基本回路（即每个结点的次数都是2），则命题得证。这是因为图是连通的，C_1 与 G_1 必然有公共结点（但无公共边），从这个公共结点出发绕行 C_1 一圈再绕行 G_1 一圈即可回到原点，显然这样绕行经过了所有的边一次而不重复，因而是欧拉回路。

如果 G_1 还不是基本回路，即仍存在次数大于2的偶次结点，则 G_1 至少含有一个基本回路，例如图(b)中的 *abcda*，用 C_2 表示，从 G_1 中删去 C_2，得到图 G_2，如图(c)所示。如果 G_2 仍不是基本回路，则必含有一个基本回路 C_3，又可仿照上述操作从 G_2 中删去 C_3，如此继续下去，由于图的有限性，最后得到的子图 G_m，它的结点次数均为2，即是一个基本回路 C_m。 ▌

上面得到的这些基本回路 $C_1, C_2, \cdots, C_m$，它们没有公共边，但至少有一公共点，因此可以把这些回路在公共点处连接起来而构成通过 G 的所有边一次且仅一次的回路，即欧拉回路。

用这一定理检查图8.2的两个图，很容易判定(a)图存在欧拉回路而(b)图则不然。

定义 8.2 若无向图 G 的每个结点次数均为偶数，则称 G 为欧拉图，简称 E 图。

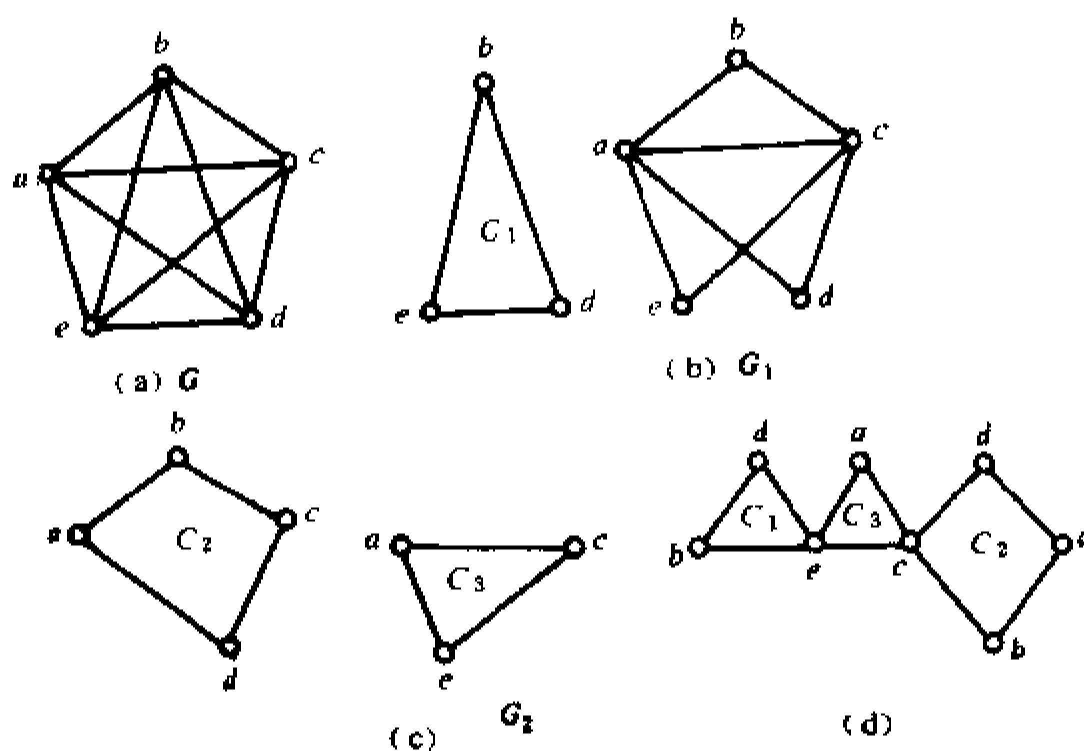


图 8.3

由定理8.1可知具有欧拉回路的图一定是欧拉图。但是由于定义8.2并未规定连通性的条件，所以欧拉图可以不是连通图，如图8.4所示就是一个非连通E图。

多数情况下，我们感兴趣的是连通E图，对非连通E图，它的每个连通分支必然含欧拉回路，可以按连通E图来处理。

一个图不是E图，但它的子图可以是E图，称为E子图。如图8.5表示图G及它的两个E子图G_1和G_2。

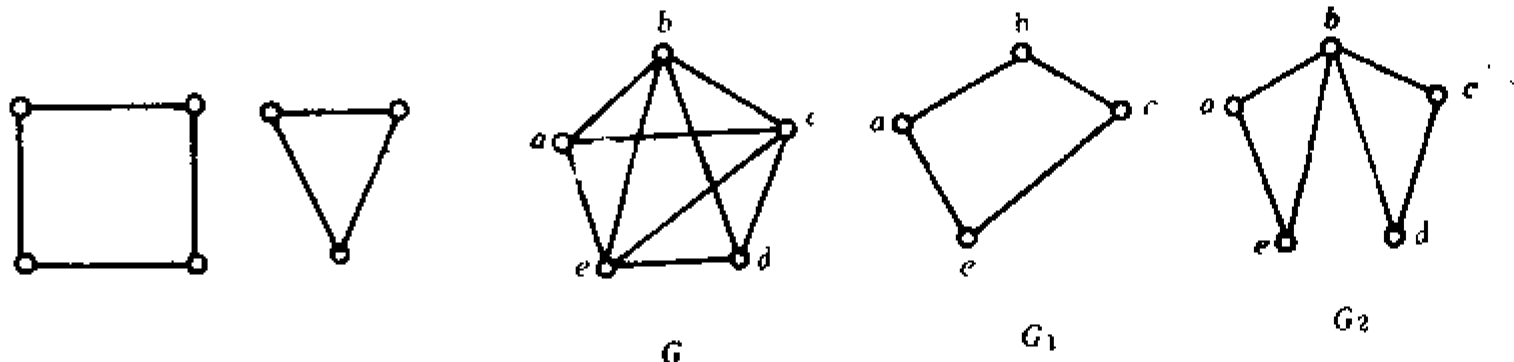


图 8.4　　图 8.5

定理8.2　若G_1和G_2是图G的E子图，则$G_1 \oplus G_2$也是G的E子图。

证：令$G'=G_1 \oplus G_2$，对G'中的任一结点v，设G_1中与v关联的边有r条，记作e_{11}，e_{12}，…，e_{1r}，而G_2中与v关联的边有s条，记作e_{21}，e_{22}，…，e_{2s}，不失一般性，设$e_{11}=e_{21}$，$e_{12}=e_{22}$，…，$e_{1t}=e_{2t}$，但$e_{1(t+1)}$，$e_{1(t-2)}$，…，e_{1r}与$e_{2(t+1)}$，$e_{2(t+2)}$，…，e_{2s}不再有相同的边，因此结点v在G'中的次数为

$$deg(v)=r+s-2t$$

因为G_1和G_2都是E图，故r和s都是偶数，则$deg(v)$必为偶数，由于v的任意性，所以G'是E图。　▌

推论：设$G_i(i=1,2,\cdots,n)$是E图，则$G_1 \oplus G_2 \oplus \cdots \oplus G_n$也是$E$图。

例如：图8.6的G'就是图8.5中两个E图G_1与G_2的环和，可见G'也是E图。

定理8.3　若P_1和P_2是同一对结点之间的两条简单路径，则$P_1 \oplus P_2$是E图。

证：设P_1和P_2是结点v_i和v_j之间的两条简单路径，我们在v_i与v_j之间连一条边e，显然$P_1 \cup \{e\}$是E图，同理$P_2 \cup \{e\}$也是E图，由定理8.2的推论知

$$[P_1 \cup \{e\}] \oplus [P_2 \cup \{e\}]=P_1 \oplus P_2$$

也是E图。　▌

推论：若$P_i(i=1,2,\cdots,2k)$是同一于结点之间的简单路径，则$P_1 \oplus P_2 \oplus \cdots \oplus P_{2k}$是$E$图

证：$P_1 \oplus P_2 \oplus \cdots \oplus P_{2k}=(P_1 \oplus P_2) \oplus (P_3 \oplus P_4) \oplus \cdots, \oplus (P_{2k-1} \oplus P_{2k})$

由本定理知，$(P_1 \oplus P_2)$，$(P_3 \oplus P_4)$，…，$(P_{2k-1} \oplus P_{2k})$都是E图，又由定理8.2的推论可知，它们的环和亦是E图。　▌

例8.1　在图8.7所示的图中，结点v_1与v_3之间的所有路径如图8.8所示，显然可得：

$$P_1 \oplus P_2=G_1$$
$$P_2 \oplus P_3=G_2$$
$$P_3 \oplus P_4=G_3$$
$$P_1 \oplus P_2 \oplus P_3 \oplus P_4=G_4$$

其中G_1，G_2，G_3，G_4如图8.9所示，可见它们都是E图；而$P_1 \oplus P_2 \oplus P_3$则不是$E$图，它

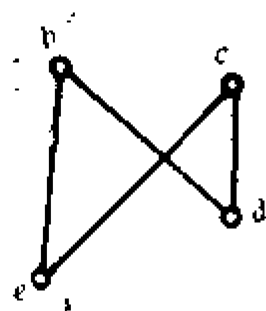


图 8.6

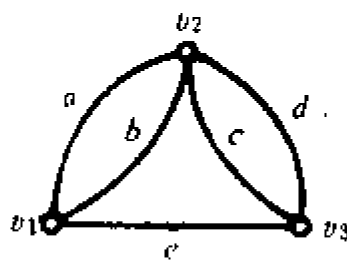


图 8.7

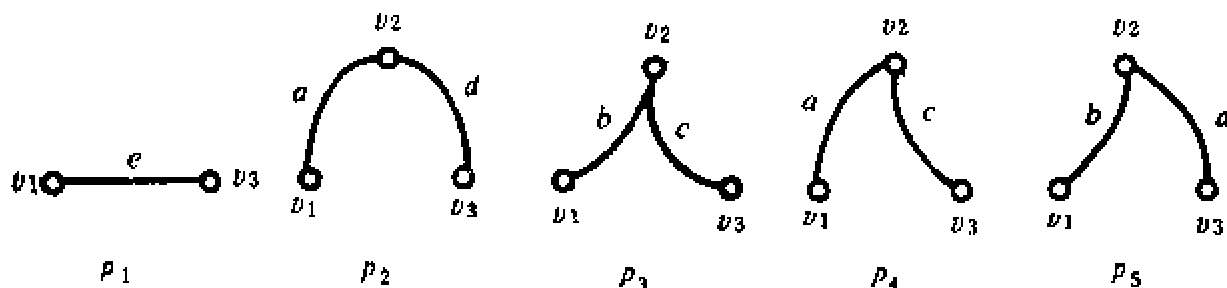


图 8.8

正是图8.7的图G。由此表明两点之间的路径数目必须是偶数，它们的环和才可能是E图。

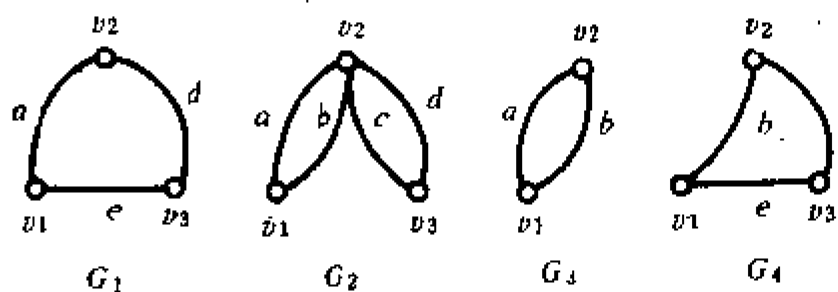


图 8.9

定义 8.3 一个不含孤点的无向图$G=(V, E)$，如果存在经过每一条边一次且仅一次的路径，则称此路径为欧拉路径。

欧拉路径问题在我国也常称为一笔画问题，即一个图形可以一笔画出，例如图8.10中的图，(a)和(b)图可以一笔画出而(c)，(d)则不可能一笔画出。

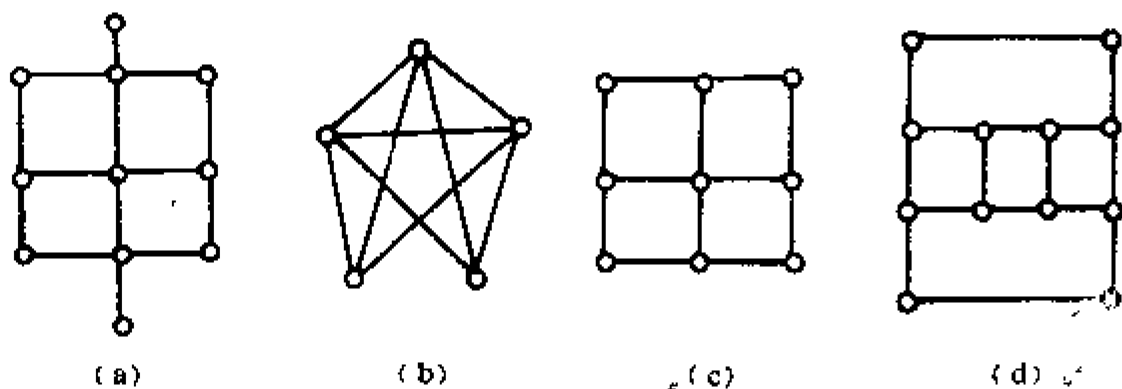


图 8.10

定理 8.4 在连通无向图$G=(V, E)$中，当且仅当结点v_i和v_j的次数为奇数，其余结点的次数均为偶数时，v_i与v_j之间才有一条欧拉通路。

证：我们在图G的结点v_i与v_j之间添上一条边(v_i, v_j)，于是图G成为图G'。

必要性：若G中从v_i到v_j存在欧拉通路，则在G'中由于添加边(v_i, v_j)而构成欧拉回路，由定理8.1知G'的所有结点次数均为偶数，去掉添加边(v_i, v_j)，G'回到G，此时v_i和v_j的次数较在G'中时少1而成为奇数，其余结点的次数不受影响，仍为偶数。

充分性：若G中除v_i和v_j的次数为奇数外其余结点均为偶次，则添上一条边(v_i, v_j)

之后得到的图 G' 是欧拉图，从 v_i 出发经过所有的边一次且仅一次（包括边 (v_i, v_j)）回到 v_i 构成欧拉回路，除掉添加边 (v_i, v_j) 后，G' 回到 G，显然从 v_i 出发仍能经过所有的边一次且仅一次而到达 v_j，故图 G 具有欧拉通路。 ▌

定义 8.4 如果图 G 只有两个结点次数为奇数，则称 G 为 M 图，并称两个奇次结点为 M 图的端点。

与 E 图类似，M 图可能是连通图，也可能是非连通图。具有欧拉路径的图一定是 M 图，可称为连通 M 图。

显然，在 M 图的两个端点之间添上一条边，得到 E 图，而删除 E 图的任意一条边，将成为 M 图。

定理 8.5 设 C 是一个简单回路，P 是一条简单路径，则 $C\oplus P$ 是 M 图

证： 设 $G'=C\oplus P$，对 G' 的任一结点 v，设 C 与 v 关联的边有 r 条，P 与 v 关联的边有 s 条，不失一般性，设其中有 t 条边是相同的，则 v 的次数为

$$deg(v)=r+s-2t$$

如果 v 不是路径 P 的两个端点，则 r 和 s 均为偶数，而 P 的两个端点次数仍为奇数，故 G' 是 M 图。如果 v 是 P 的端点，则 r 是偶数 s 是奇数，故 v 的次数仍为奇数，G' 也是 M 图。 ▌

推论： 若 M 是 M 图，E 是 E 图，则 $M\oplus E$ 是 M 图。

定理 8.6 若 M_1 和 M_2 是有相同端点的 M 图，则 $M_1\oplus M_2$ 是 E 图。

证： 在 M_1 和 M_2 的公共端点之间增添一条边 e，则 $M_1\cup\{e\}$ 和 $M_2\cup\{e\}$ 都是 E 图，由定理 8.2 的推论可知

$$[M_1\cup\{e\}]\oplus[M_2\cup\{e\}]=M_1\oplus M_2$$

是 E 图。 ▌

§8.2 欧拉图的寻迹

一个具有欧拉回路的连通无向图，虽然一定可以从图的任一结点出发，遍历每条边一次且仅一次而又回到原点，但是，如果遍历的路径不得法也可能无法实现。如图 8.11 所示的图是欧拉图，我们从 a 点出发，如果行走的路径为 $a\to b\to c\to a$，即经过边 (a,b)，(b,c)，和 (c,a)，这时必须重复已走过的边才能遍历其他边。由此说明欧拉图的寻迹也存在一个正确的走法即算法问题。

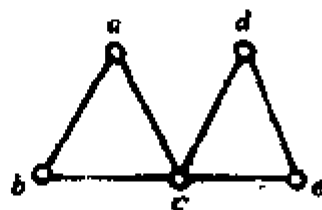


图 8.11

定理 8.1 的证明，实际上也提供了一种寻找欧拉回路的方法，即找出图的所有基本回路，然后按它们的公共点把这些基本回路连接起来，构成一个大回路，即是欧拉回路。

下面介绍一个较好的算法。它是由 Fleury 提出的，算法不需要重复地计算图的基本回路，而是用依次访问一条边的方法来构造欧拉回路，算法的思路是这样的：每当访问一条边时，先要进行检查，如果可供访问的边不止一条，则应选一条不致破坏未访问的边构成的子图的连通性的边作为访问边，即不要选择这个子图的割边作为访问边。

例如图 8.11，我们从 a 点出发依次访问了边 (a, b) 和 (b, c) 而到达 c 点，这时下一步访问的边有三条可供选择，即边 (c, a)，(c, d) 和 (c, e)，对未访问过的边构成的子图

来说，(c, a) 是割边而 (c, d) 和 (c, e) 则不是割边，如果选择 (c, a) 为访问边，必然破坏子图的连通性而使遍历图失败，如果选择 (c, d) 或 (c, e) 作为访问边，则访问可以连续进行最终构成欧拉回路。为此，算法给出了如下参变量：

w：访问的起始点（可任选图的一点）

EC：顺序通过的结点集合

E'：已通过的边集合

CV：当前访问的结点

$A(v)$：结点 v 在子图 $(G-E')$ 中的邻接表

Fleury 算法：

1. $EC \leftarrow \{w\}$
2. $CV \leftarrow w$
3. $E' \leftarrow \phi$
4. While $|A(w)|>0$ *do*

   begin

5. if $|A(CV)|>1$ then
6. 找一点 $v \in A(CV)$ 且 (CV, v) 不是子图 $(G-E')$ 的割边
7. else 在 $A(CV)$ 中这一结点记作 v
8. 在 $A(CV)$ 中删除 v，$A(v)$ 中删除 CV
9. $E' \leftarrow E' \cup \{(CV, v)\}$
10. $CV \leftarrow v$
11. 将 CV 加到 EC 的表末尾

    end

12. 输出 EC

算法的计算量主要为执行 *While* 循环语句，从第 5 行到第 11 行需要重复 $|E|$ 次，在执行第 6 行语句时，即判定访问的边是否是 $(G-E')$ 的割边，计算量为 $O(|E-E'|)$，因此总的计算量是 $O(|E|^2)$ 级。

例 8.2 用上述算法求出图8.12所示的欧拉图的欧拉回路。

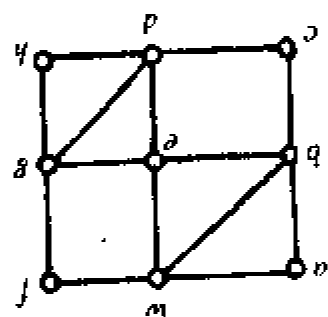


图 8.12

解：初始状态，

出发点：w，$EC \leftarrow \phi$，$E' \leftarrow \phi$，$|A(w)|=4$，$|A(a)|=2$，$|A(b)|=4$，$|A(c)|=2$，$|A(d)|=4$，$|A(e)|=4$，$|A(f)|=2$，$|A(g)|3$，$|A(h)|2$

开始，$CV \leftarrow w$，$EC \leftarrow \{w\}$

1）因 $|A(CV)|=|A(w)|=4>1$

找到点 a 且 (w, a) 不是 $G-E'$ 的割边，于是

$A(w) \leftarrow \{a, b, e, f\}-\{a\}=\{b, e, f\}$

$A(a) \leftarrow \{w, b\}-\{w\}=\{b\}$

$E' \leftarrow \phi \cup \{(w, a)\}=\{(w, a)\}$

$CV \leftarrow a$，$EC \leftarrow \{w, a\}$

2）因 $|A(CV)|=|A(a)|=|\{b\}|=1$，故 b 记作 v

$A(a) \leftarrow \{b\} - \{b\} = \phi$

$A(b) \leftarrow \{a, c, e, w\} - \{a\} = \{c, e, w\}$

$E' \leftarrow \{(w, a)\} \cup \{(a, b)\} = \{(w, a), (a, b\}$

$CV \leftarrow b, \quad EC \leftarrow \{w, u, b\}$

3）因$|A(CV)| = |A(b)| > 1$

找到点 c 且（b，c）不是 $G - E'$ 的割边，于是

$A(b) \leftarrow \{c, e, w\} - \{c\} = \{e, w\}$

$A(c) \leftarrow \{b, d\} - \{b\} = \{d\}$

$E' \leftarrow \{(w, a), (a, b), (b, c)\}$

$CV \leftarrow C, \quad EC \leftarrow \{w, a, b, c\}$

4）因$|A(CV)| = |A(c)| = |\{d\}| = 1$，故 d 记作 v

$A(c) \leftarrow \{d\} - \{d\} = \phi$

$A(d) \leftarrow \{c, e, g, h\} - \{c\} = \{e, g, h\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d)\}$

$CV \leftarrow d, \quad EC \leftarrow \{w, a, b, c, d\}$

5）因$|A(CV)| = |A(d)| > 1$

找到点 e 且（d，e）不是 $G - E'$ 的割边，于是

$A(d) \leftarrow \{e, g, h\} - \{e\} = \{g, h\}$

$A(e) \leftarrow \{d, b, g, w\} - \{d\} = \{b. g, w\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e)\}$

$CV \leftarrow e, \quad EC \leftarrow \{w, a, b, c, d, e\}$

6）因$|A(CV)| = |A(e)| > 1$

找到点 b 且（e, b）不是 $G - E'$ 的割边，于是

$A(e) \leftarrow \{b, g, w\} - \{b\} = \{g, w\}$

$A(b) \leftarrow \{e, w\} - \{e\} = \{w\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e), (e, b)\}$

$CV \leftarrow b, \quad EC \leftarrow \{w, a, b, c, d, e, b\}$

7）因$|A(CV)| = |A(b)| = |\{w\}| = 1$，故 w 记作 v，

$A(b) \leftarrow \{w\} - \{w\} = \phi$

$A(w) \leftarrow \{b, e, f\} - \{b\} = \{e, f\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e), (e, b), (b, w)\}$

$CV \leftarrow w, EC \leftarrow \{w, a, b, c, d, e, b, w\}$

8）因$|A(CV)| = |A(w)| > 1$

找到点 e 且（w，e）不是 $G - E'$ 的割边，于是

$A(w) \leftarrow \{e, f\} - \{e\} = \{f\}$

$A(e) \leftarrow \{g, w\} - \{w\} = \{g\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e), (e, b), (b, w), (w, e)\}$

$CV \leftarrow e, EC \leftarrow \{w, a, b, c, d, e, b, w, e\}$

9）因$|A(CV)| = |A(e)| = |\{g\}| = 1$，故 g 记作 v

$A(e) \leftarrow \{g\} - \{g\} = \phi$

$A(g) \leftarrow \{e, f, d, h\} - \{e\} = \{f, d, h\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e), (e, b), (b, w), (w, e), (e, g)\}$

$CV \leftarrow g$, $EC \leftarrow \{w, a, b, c, d, e, b, w, e, g\}$

10）因$|A(CV)| = |A(g)| > 1$

找到点f，但 (e, f) 是 $G-E'$ 的割边，故另找一点d，(g, d) 不是 $G-E'$ 的割边，于是

$A(g) \leftarrow \{f, d, h\} - \{d\} = \{f, h\}$

$A(d) \leftarrow \{g, h\} - \{g\} = \{h\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e), (e, b), (b, w), (w, e), (e, g), (g, d)\}$

$CV \leftarrow d$, $EC \leftarrow \{w, a, b, c, d, e, b, w, e, g, d\}$

11）因$|A(CV)| = |A(d)| = \{h\}| = 1$，故 h 记作 v

$A(d) \leftarrow \{h\} - \{h\} = \phi$

$A(h) \leftarrow \{d, g\} - \{d\} = \{g\}$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d,e),(e,b),(b,w),(w,e),(e,g),(g,d),(d,h)\}$

$CV \leftarrow h$, $EC \leftarrow \{w, a, b, c, d, e, b, w, e, g, d, h\}$

12）因$|A(CV)| = |A(h)| = |\{g\}| = 1$，故 g 记作 v

$A(h) \leftarrow \{g\} - \{g\} = \phi$

$A(g) \leftarrow \{f, h\} - \{h\} = \{f\}$

$E' \leftarrow \{(w,a),(a,b),(b,c),(c,d).(d,e),(e,b),(b,w),(w,e),(e,g),(g,d),(d,h),(h,g)\}$

$CV \leftarrow g$, $EC \leftarrow \{w, a, b, c, d, e, b, w, e, g, d, h, g\}$

13）因$|(CV)| = |A(g)| = |\{f\}| = 1$，故 f 记作 v

$A(g) \leftarrow \{f\} - \{f\} = \phi$

$A(f) \leftarrow \{g, w\} - \{g\} = \{w\}$

$E \leftarrow \{(w,a),(a,b),(b,c),(c,d),(d,e),(e,b),(b,w),(w,e),(e,g),(g,d),(d,h),(h,g),$
$(g, f)\}$

$CV \leftarrow f$, $EC \leftarrow \{w, a, b, c, d, e, b, w, e, g, d, h, g, f\}$

14）因$|A(CV)| = |A(f)| = |\{w\}| = 1$，故 w 记作 v

$A(f) \leftarrow \{w\} - \{w\} = \phi$

$A(w) \leftarrow \{f\} - \{f\} = \phi$

$E' \leftarrow \{(w, a), (a, b), (b, c), (c, d), (d, e), (e, b), (b, w), (w, e), (e, g), (g,d),$
$(d, h), (h, g), (g, f), (f, w)\}$

$CV \leftarrow w$, $EC \leftarrow \{w, a, b, c, d, e, b, w, e, g, d, h, g, f, w\}$

15）因$|A(w)| = 0$，计算结束，输出 EC

对 Fleury 算法的正确性，证明如下：

首先算法是行得通的，因为在初始状态下，算法的第 4 行$|A(w)| > 0$是存在的，同时算法的第 5 行$|A(CV)| > 1$也是存在的。开始之后$|A(CV)|$有两种可能：

（1）$|A(CV)| = 1$，意味着与访问点 CV 关联的边只有一条，即可执行算法第 7 行

（2）$|A(CV)| > 1$，由于 CV 是访问点，它在 $G-E'$ 中的次数必为奇数。这时执行

算法第 6 行，要求找一个不是 $G-E'$ 割边的邻接点，下面证明：当 $|A(CV)|>1$ 时，与访问点 CV 关联的边最多有一条是 $G-E'$ 的割边，其余都是回路边，因此找一条不是割边而是回路边是完全可能的，而去掉回路边不影响图的连通性，所以算法可以继续下去直到访问完所有的边为止。

下面证明当 $|A(CV)|>1$ 时，与 CV 关联的边最多有一条是 $G-E'$ 的割边

用反证法，设与 CV 关联的边有两条是割边，如图 8.13 所示，设 e_i 和 e_j 都是割边，则去掉 e_i 后，e_i 的另一端点 v_i 与点 CV 不再连通，故 CV 不会在包含 v_i 的连通子图 G_i 中，同理，去掉割边 e_j 后，CV 与 e_j 的另一端点 v_j 亦不连通，则 CV 也不会在包含 v_j 的连通中子 G_j 图。

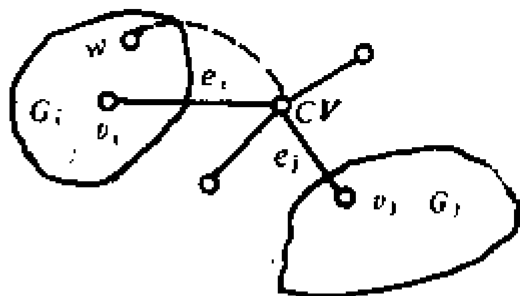


图 8.13

由于 e_i, e_j 是割边，各自不能形成回路，因此去掉 e_i, e_j 后，v_i 与 v_j 将不连通，所以 G_i 与 G_j 是不连通的，在 G_i 中，v_i 已是奇次结点（v_i 原是偶次结点，因去掉边 e_i 后成为奇次结点），根据任一连通图奇次结点的数目必为偶数这一定理，在 G_i 中必须还有 $G-E$ 的一个奇次结点，同理，在 G_j 中 v_j 已是奇次结点，必须还有 $G-E'$ 的另一个奇次结点。

但是在 $G-E'$ 中只有两个奇次结点，一个是起始点 w，另一个则是当前访问点 CV，前面已证明 CV 不在 G_i 也不在 G_j 中，而 w 也不可能同时在 G_i 和 G_j 中，因而出现矛盾，因此不可能存在两条割边，更不可能存在多于两条的割边。最多只有一条割边，即寻找一条非割边是完全可能的。 ▮

在执行算法第 6 行时需要判断割边，因此还需要设计一个判断割边的子程序，可以用判断图的连通性来判断割边，也可采用别的算法这里就不一一细述了。

§8.3 中国邮路问题

一个邮递员从邮局出发，到所辖街道投递邮件，最后返回邮局，如果他必须走遍所辖的每条街道至少一次，那么应如何选择投递路线，使所走的路程最短，这个问题首先是由我国学者管梅谷教授于 1962 年提出的，因此称为中国邮路问题。

中国邮路问题具有普遍意义，如公安执勤人员执行巡逻任务、企业的巡回检测等等都存在如何选择最短巡回路线的问题。

如果把投递区的街道用一条边 (v_i, v_j) 表示，街道的长度用边权 $w(v_i, v_j)$ 表示，邮局、街道交叉口用点表示，那么一个投递区构成一个边权连通无向图，邮路问题用图论的语言来描述，就是在一个边权连通无向图中，怎样寻找一个回路C，使得C经过每条边至少一次且C的长度最短。

因此邮路问题，既是一个与欧拉回路有关，而且也与最短路径有关的问题。下面我们分三种情况进行讨论

一、最理想情况下邮路问题的解

如果邮路 $G=(V, E)$ 是一个连通欧拉图，则存在欧拉回路，这是最理想情况，这时

每条边只需经过一次，回路的长度

$$L(C)=\Sigma w(v_i, v_j)$$
$$(v_i, v_j)\in E$$

是最小可能值。邮路的投递路线，也就是欧拉回路的遍历，上一节已介绍了它的算法。

二、次理想情况下邮路问题的解

如果邮路G具有欧拉通路，即只有两个结点v_i和v_j是奇次结点，这时从v_i出发可以经过每条边一次且仅一次而到达v_j，从v_j回到v_i必须重复经过一些边，显然，要想使总的投递路线最短，必须使重复边的总长度最小，因而成为寻找v_i和v_j之间的最短路径问题，这种情况称为次理想情况。

次理想情况的算法如下

1）求出奇次结点v_i和v_j之间的最短路径P。

2）令$G^*=G\cup P$

3）G^*为E图，穿行G^*的欧拉回路就是邮路投递路线的最优解。

例 8.3 图8.14表示一邮递路线图，求出最优投递路线。

解：求出v_1到v_6的最短路径

$$P=v1,\ v4,\ v5,\ v2,\ v6$$

P的长度为6，如图8.15（a），将P加到G上，得到图G^*，如图8.15（b），最优投递路线即为G^*的欧拉回路

$$C=v_1v_2v_3v_5v_2v_6v_4v_5v_2v_6v_5v_4v_3v_1v_4v_1$$

其总长度为

$$L(C)=\sum_{(v_i,v_j)\in E^*} w(v_i,v_j)=33+6=39$$

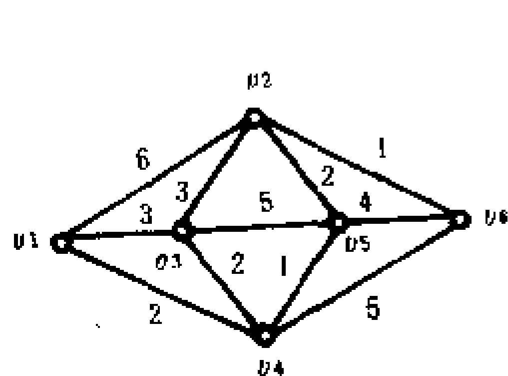
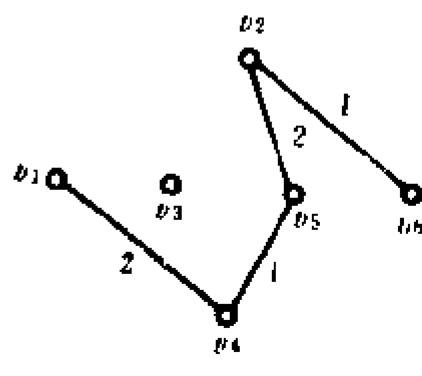
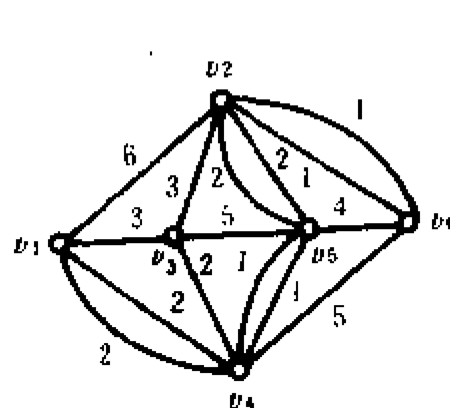


图 8.14



（a）P



（b）G^*

图 8.15

三、一般情况下邮路问题的解

当邮路既非欧拉回路，又不具有欧拉通路时，即为一般情况，在这种情况下，必须重复更多的边，但应重复哪些边才能使投递路线的总长最小呢？管梅谷教授提出了下面一个定理，规定了两个最优路线的条件。

定理 8.7 设G^*是包含了连通边权无向图G的所有边的一个回路，则G^*具有最小长度的充分必要条件是

（1）G的每条边最多重复一次

（2）在G的每个回路上，有重复边的长度和，不超过回路长度的一半。

证：必要性。首先设有些边重复n（$n\geqslant 2$）次得到的图是欧拉图，如图8.16（a）所示，（图中重复边用虚线表示）那么将重复的边数去掉偶数条（最后必然只重复一次或零次），重复边的两个端点的次数仍然是偶数，结果仍然是欧拉图，如图8.16（b）所示，所以最优的情况下边的重复不会多于一次。

其次，我们考察G的任一回路，如图8.17（a），如果把重复的边都不重复，不重复的边都重复一次，则图（a）变成图（b），这样每个结点的次数都改变了0或2，这种改变的结果并不影响结点的偶次性，图仍然保留欧拉图的性质，如果（a）重复边的长度和超过回路总长的一半，则（b）重复边的长度和必小于回路总长的一半，所以（b）是最优的。

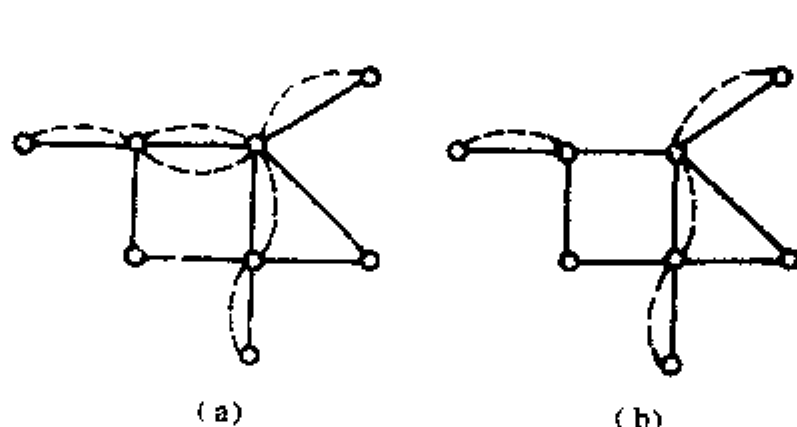

(a)　(b)

图 8.16

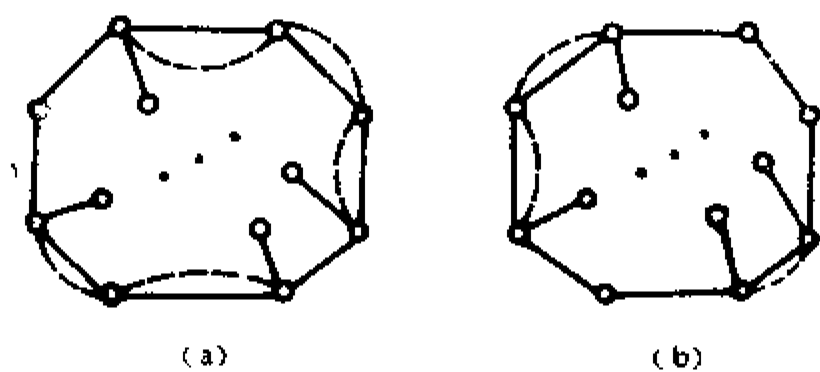

(a)　(b)

图 8.17

充分性，只要证明满足定理的两个条件的所有回路的长度均相等，即重复边的长度和相等即可。

设G_1和G_2是满足上述两个条件的回路，C是G_1与G_2中重复边的环和，由边集C组成的网络的每个分支都是欧拉回路，因为对G的任一结点v_i，G_1和G_2中与v关联的重复边的数目的奇偶性是相同的，它取决于v原来次数的奇偶性，如v原为奇次结点，则重复边的数目也为奇数，否则为偶数，因此C是回路的边不相重的并集，但在每一个回路上，G_1和G_2的重复边长度都不超过回路长的一半，故只能相等且等于回路长的一半，因而C在G_1和G_2中重复边的长度相等，于是G_1与G_2中重复边的长度相等，即G_1与G_2的长度相等。 ▌

定理的证明，为我们提供了一般情况下寻求最优投递路线的方法，称为奇偶点作业法。

奇偶点作业法

1．任给一个初始方案，使网络各结点的次数均为偶数，网络成为边权欧拉图。

2．检查每一回路重复边的长度和是否不超过回路长度的一半，如是，则现行方案即为最优解，否则进行下一步。

3．调整重复边，即将回路中重复的边改为不重复，不重复的边改为重复，返回第2步。

例 8.4 用奇例点作业法，对图8.18所示的网络，求最优投递路线

解 在图G上增添两条边（a，b），（c，d），得到一个欧拉图G_1如图8.19（a）所示。

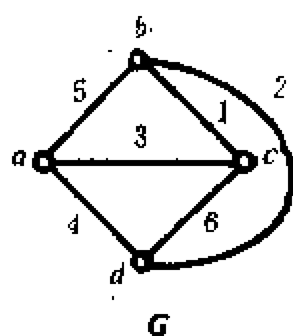


图 8.18

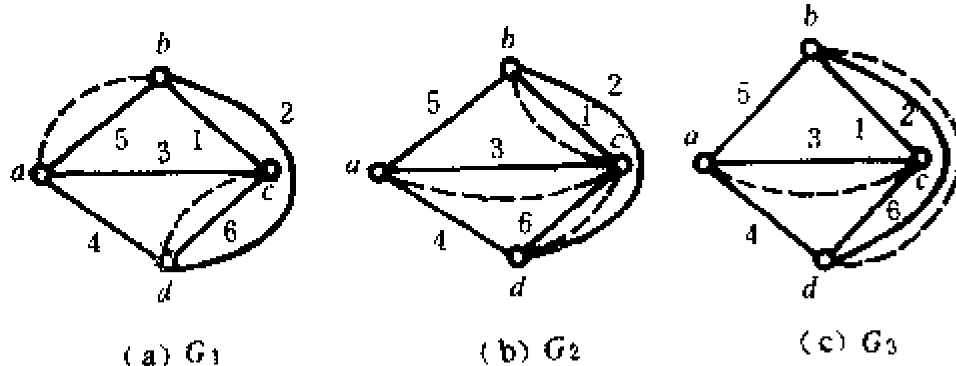


图 8.19

检查回路$abca$，重复边的长度超过回路总长的一半，调整重复边，欧拉图 G_1 变成欧拉图 G_2，如图（b）所示。

检查回路 $bcdb$，重复边的长度和超过回路长度一半，调整重复边，得到欧拉图 G_3，如图（c）所示。

检查图 G_3，不存在重复边长度超过回路长度一半的回路，因此 G_3 即是最优解。

奇偶点作业法当图的规模较大时，要检查每一回路是相当困难的，而且重复边调整后，又会出现新的需要检查的回路，这些回路的出现并无一定的规律可以遵循，如果调整一次重复边都要逐一检查每个回路，工作量之大是可想而知的。1965 年 Edmonds 提出了一个有效的算法，使中国邮路问题得到了较好解决。

算法的基本思想是：利用奇次结点之间的最小权完美匹配来确定重复边，从而使得到的欧拉图具有最优的投递路线。

我们知道，要使一般图成为欧拉图，必须使奇次结点成为偶次结点，因此须要添加重复边，奇次结点一定与重复边关联，而奇次结点的数目一定是偶数，可以把奇次结点两个一对两个一对地分成若干点对，如果这些点对之间的路径长度总和为最小，那么把这些路径所经过的边作为重复边加到原图上，不仅使图成为欧拉图，而且重复边的总长也是最小的，显然得到的这个图就是所求的最优解。而求点对之间的最短路径，实质上就是求最小权完美匹配，因此得到算法如下：

1）求出图 G 所有奇次结点之间的最短路径和距离。

2）以 G 的所有奇次结点为结点（必为偶数），以它们之间的距离作为结点之间的边的权，得到一个m阶完全图 G_1（m为奇次结点的数目）

3）求出 G_1 的最小权完美匹配 M

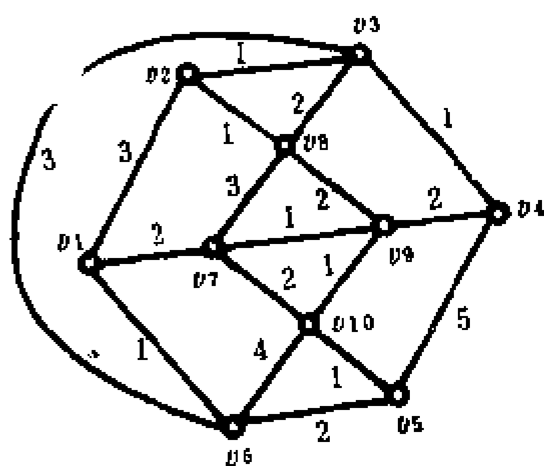


图 8.20

4）将 M 中的匹配边 (v_i, v_j) 写成 v_i 和 v_j 之间最短路径所经过的边的集合 E_{ij}

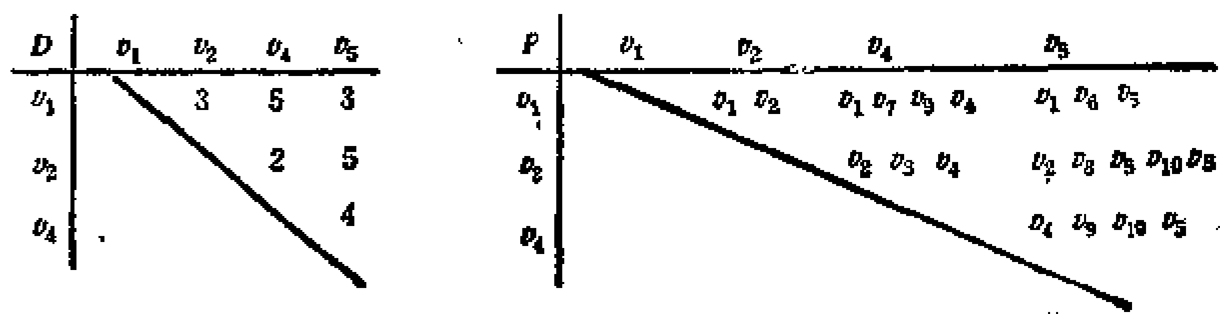
5）令 $G^*=G\cup\{E_{ij}|(v_i, v_j)\in M\}$，则 G^* 是欧拉图，即是邮路的最优解。

执行算法的第一步，可以应用弗洛伊得算法求出任意两点之间的距离和最短路径。执行算法的第 3 步，可以用求最大权匹配的算法求图 G 的最小权匹配，方法是构造一个结点和边与 G 完全相同的图 G'，G' 每条边的权等于一个非常大的数减去 G 中与之对应的边的权，则 G' 的最大权匹配对应于 G 的最小权匹配。

例 8.5 求图8.20所示的邮路的最优解。

解：图G有4个奇次结点：v_1，v_2，v_4，v_5，用弗洛伊得算法求出它们之间的距离和最短路径如下：

以v_1，v_2，v_4，v_5为结点，它们之间的距离作为边的权构造完全图G_1，如图8.22(a)所示。取最大数10构造与G_1对应的图G_1'，如图（b）所示。



D	v_1	v_2	v_4	v_5
v_1		3	5	3
v_2			2	5
v_4				4

P	v_1	v_2	v_4	v_5
v_1		$v_1\ v_2$	$v_1\ v_7\ v_9\ v_4$	$v_1\ v_6\ v_5$
v_2			$v_2\ v_3\ v_4$	$v_2\ v_8\ v_5\ v_{10}\ v_5$
v_4				$v_4\ v_9\ v_{10}\ v_5$

图 8.21

求G_1'的最大权匹配（即G_1的最小权匹配）得

$$M=\{(v_1,v_5),(v_2,v_4)\}$$

将匹配边换成两点间最短路径的边集

$$E_M=\{(v_1,v_6),(v_6,v_5),(v_2,v_3),(v_3,v_4)\}$$

将边集E_M作为重复边加到G上得到欧拉图

$$G^*=(V,E^*)=G\cup E_M$$

如图8.23所示。则G^*即是邮路的最优解。

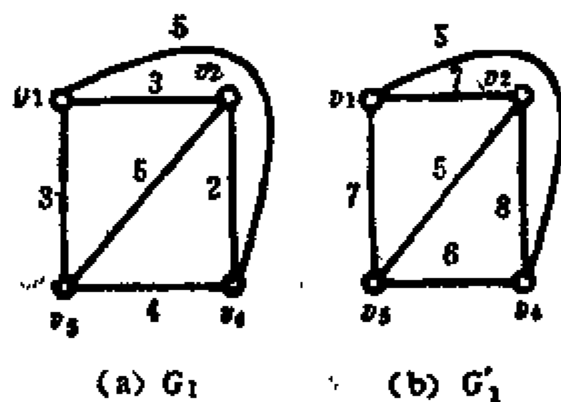


图 8·22

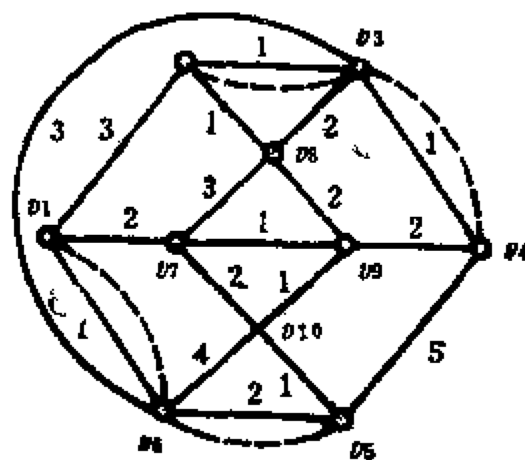


图 8.23

§8.4 有向欧拉图

定义 8.5 一个连通有向图$D=(V,E)$，如果

（1）存在沿着边的方向经过每条边一次且仅一次的路径，则称此路径为有向欧拉路径。

（2）存在沿着边的方向经过每条边一次且仅一次的回路，则称此回路为有向欧拉回路。

（3）具有有向欧拉回路的有向图，称为有向欧拉图。

如图8.24（a）所示的有向图具有有向欧拉路径，而（b）所示的图则是一个有向欧

拉图。

一个有向图D，如果每一结点的引入次数都与它的引出次数相等，则称D是平衡的。

与定理 8.1 及定理 8.4 类似，我们得到有向图的如下定理，定理的证明方法也是相似的，这里不再赘述。

定理 8.8 对有向图$D=(V,E)$

（1）D是有向欧拉图当且仅当D是连通且是平衡的。

（2）D有有向欧拉路径当且仅当D是连通的且它的结点次数满足：

$$1° \quad \deg^+(v)=\deg^-(v), \text{对所有的 } v\neq v_1 \text{ 或 } v_2$$

$$2° \quad \deg^+(v_1)=\deg^-(v_1)+1$$

$$3° \quad \deg^-(v_2)=\deg^+(v_2)+1$$

与无向欧拉图的寻迹一样，寻找有向欧拉图的欧拉回路，也存在一个正确的走法问题。下面介绍的有向欧拉图寻迹算法，它的基本思路是：先构造一棵有向欧拉图的生成树T，然后采取逆向寻迹的方法获得有向欧拉回路。

如图8.25（a）表示一个有向欧拉图，以任一结点（比如结点u）为根，采用深度优先搜索法（DFS算法）得到图的一棵生成树T，如图（b）所示。于是有如下定理

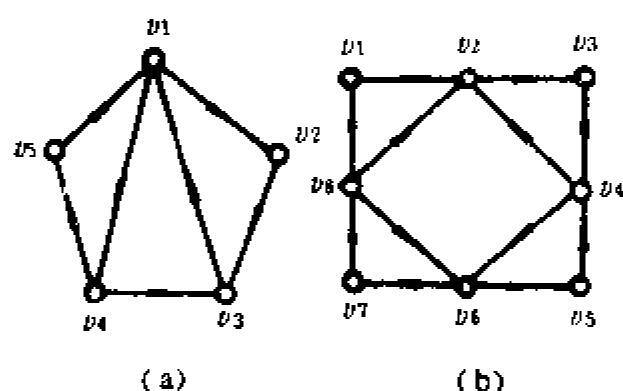


图 8.24

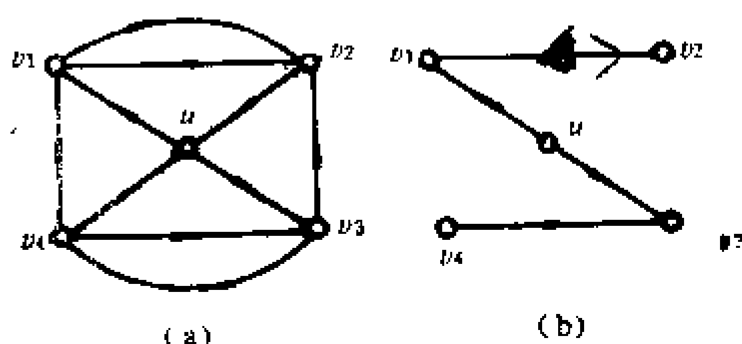


图 8.25

定理 8.9 设D是连通、平衡的有向图，T是D的一棵以结点u为根的生成树，则有向欧拉回路可以由下面的逆向寻迹法获得

（1）逆向寻迹的起始边是与u关联的任一条边。

（2）选取与当前结点关联的逆向边作为后继边，并且满足：

（a）任何边不得重复

（b）若与当前结点关联的逆向边中存在不属于T的逆向边时，不得选取属于T的逆向边。

（3）当寻迹到达某一结点时，与该结点关联的边都已选取过，寻迹过程结束。

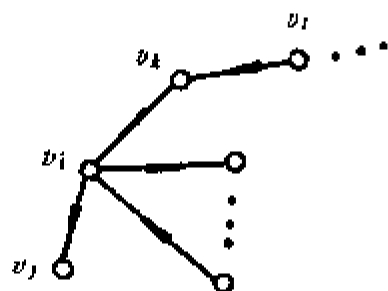


图 8.26

证：由于D是平衡的，按上述规则寻迹的路径只能终止于结点u，从面形成一条回路，假设这一回路并未含有D的某一条边(v_i,v_j)，那么由于D是平衡的，则一定存在一条以v_i为末端的边(v_K,v_i)未选取过，如图8.26所示。根据规则（2）（b），这条边可以当作为属于T的逆向边未被选取，同理，必然存在一条以v_K为末端的边(v_t, v_K)未被选取，依此类推，我们将得到一条逆向的路径回溯到结点u，由于D是平衡的，说明u还存在一条逆向边未被选取，这与步骤（3）相矛盾，因此这一回路

是有向欧拉回路。 ▌

有向欧拉图寻迹算法

1. 求出图$D=(V,E)$以结点u为根的生成树T，并给出如下赋值：如边$\langle v_i,v_j\rangle \in T$则令$T(\langle v_i,v_j\rangle)$为真，否则为假。
2. for 每一结点$v\in V$ *do*
   begin
3. $Av\leftarrow\phi$
4. $I(v)\leftarrow O$
   end
5. for 每一条边$\langle v_i,v_j\rangle\in E$ *do*
   if $T(\langle v_i,v_j\rangle)$ ***then*** 将v_i加到Av_j的尾端
   else 将v_i加到Av_j的首端
6. $EC\leftarrow\phi$
7. $CV\leftarrow u$
8. While $I(CV)\leqslant deg^-(CV)$ do
   begin
9. 将CV加到EC的首端
10. $I(CV)\leftarrow I(CV)+1$
11. $CV\leftarrow A_{CV}(I(CV))$
    end
12. 输出EC

算法说明：

算法的第一行首先用DFS算法求出有向图D以结点u（可以是任意点）为根的生成树T，并设置一布尔变量$T(\langle v_i,v_i\rangle)$，若边$\langle v_i,v_j\rangle\in T$，其值为真，否则为假。$Av$是结点$v$的逆向边邻接结点表集，表集中结点排列的前后次序由第5行给出，以图8.25为例，与结点v_1关联的逆向边为$\langle u,v_1\rangle$和$\langle v_2,v_1\rangle$，因$\langle u,v_1\rangle\in T$，故邻接结点$u$排在表集$Av_1$的尾端，而$\langle v_2,v_1\rangle\notin T$，故邻接结点$v_2$排在表集$Av_1$的首端，于是得到$Av_1=[v_2u]$。$I(v)$为结点$v$的表集$Av$中结点的序号，若$I(v)=K$，即表示$Av$中从头数起第$K$位。例如：若$I(v_1)=1$，则$Av_1(I(v_1))=v_2$，$CV$为当前访问的结点，$EC$为寻迹产生的路径，最后得到的$EC$即为有向欧拉回路。

由算法可知，执行第一行，用DFS算法求有向图的生成树计算量为$O(\max(n,|E|))$，对于有向欧拉图，都有$|E|\geqslant n$，所以第1行的计算量为$O(|E|)$时间，执行第5行的条件语句时，对每一条边给出末端点的邻接表，总的边数为$|E|$，所以计算量也是$O(|E|)$级的。执行第2行语句时给每一结点的Av和$I(v)$赋初值，需要进行$O(n)$次，从第8行到第11行为对回路的寻迹，执行这一循环体需要的时间为$O(|E|)$，因此总的计算量是$O(|E|)$级的。

例 8.6 试用上述算法求图8.25（a）的欧拉回路。设已求出图的一棵生成树如图（b）所示。

解：1. $T(\langle u,v_1\rangle)=true$,　$T(\langle v_1,v_2\rangle)=true$,

$T(\langle u,v_3\rangle)=true$,　$T(\langle v_3,v_4\rangle)=true$,

其余　$T(\langle v_i,v_j\rangle)=false,\langle v_i,v_j\rangle\notin T$

2. $Av_1=[v_2,u]$,　$Av_2=[v_3,v_1]$,　$Av_3=[v_4,u]$,　$Av_4=[v_1,v_3]$,　$Au=[v_2,v_4]$,

3. 对所有 $v\in V, deg^-(v)=2, A_v\leftarrow\phi, I(v)\leftarrow O, EC\leftarrow\phi$

4. $CV\leftarrow u$，因 $I(u)=0\leqslant deg^-(u)$，故

$$EC\leftarrow\{u\}$$
$$I(u)\leftarrow 1$$
$$CV\leftarrow A_u(1)=v_2$$

5. 因 $I(CV)=I(v_2)\leqslant deg^-(v_2)$，故

$$EC\leftarrow\{v_2\}\cup EC=\{v_2,u\}$$
$$I(v_2)\leftarrow 1$$
$$CV\leftarrow Av_2(1)=v_3$$

6. 因 $I(CV)=I(v_3)\leqslant deg^-(v_3)$，故

$$EC\leftarrow\{v_3\}\cup EC=\{v_3,v_2,u\}$$
$$I(v_3)\leftarrow 1$$
$$CV\leftarrow Av_3(1)=v_4$$

7. 因 $I(CV)=I(v_4)\leqslant deg^-(v_4)$，故

$$EC\leftarrow\{v_4\}\cup EC=\{v_4,v_3,v_2,u\}$$
$$I(v_4)\leftarrow 1$$
$$CV\leftarrow Av_4(1)=v_1$$

8. 因 $I(CV)=I(v_1)\leqslant deg^-(v_1)$，故

$$EC\leftarrow\{v_1\}\cup EC=\{v_1,v_4,v_3,v_2,u\}$$
$$I(v_1)\leftarrow 1$$
$$CV\leftarrow Av_1(1)=v_2$$

9. 因 $I(CV)=I(v_2)=1\leqslant deg^-(v_2)$，故

$$EC\leftarrow\{v_2\}\cup EC=\{v_2,v_1,v_4,v_3,v_2,u\}$$
$$I(v_2)\leftarrow 1+1=2$$
$$CV\leftarrow Av_2(2)=v_1$$

10. 因 $I(CA)=I(v_1)=1\leqslant deg^-(v_1)$，故

$$EC\leftarrow\{v_1\}\cup EC=\{v_1,v_2,v_1,v_4,v_3,v_2,u\}$$
$$I(v_1)\leftarrow 1+1=2$$
$$CV\leftarrow Av_1(2)=u$$

11. 因 $I(CA)=I(u)=1\leqslant deg^-(u)$，故

$$EC\leftarrow\{u\}\cup EC=\{u,v_1v_2,v_1,v_4,v_3,v_2,u\}$$
$$I(u)\leftarrow 1+1=2$$
$$CV\leftarrow Au(2)=v_4$$

12. 因 $I(CV)=I(v_4)=1\leqslant deg^-(v_4)$，故

$$EC\leftarrow\{v_4\}\cup EC=\{v_4,u,v_1,v_2,v_1,v_4,v_3,v_2,u\}$$

$$I(v_4)\leftarrow 1+1=2$$

$$CV\leftarrow Av_4(2)=v_3$$

13．因 $I(CV)=I(v_3)=1\leqslant deg^-(v_3)$，故

$$EC\leftarrow\{v_3\}\cup EC=\{v_3,v_4,u,v_1,v_2,v_1,v_4,v_3,v_2,u\}$$

$$I(v_3)\leftarrow 1+1=2$$

$$CV\leftarrow Av_3(2)=u$$

14．因 $I(CV)=I(u)=2\leqslant deg^-(u)$，故

$$EC\leftarrow\{u\}\cup EC=\{u,v_3,v_4,u,v_1,v_2,v_1,v_4,v_3,v_2,u\}$$

$$I(u)\leftarrow 2+1=3$$

$$CV\leftarrow Au(3)=\phi$$

15．因 $I(CV)\nleqslant deg^-(CA)$，计算结束，输出

$$EC=\{u,v_3,v_4,u,v_1,v_2,v_1,v_4,v_3,v_2,u\}$$

则有向欧拉回路的寻迹为从 EC 的首端依次访问各结点直到末端所形成的有向回路。从步骤 4 到15的迭代过程可列成表如表 8.1 所示。

表 8.1

迭代次数	CV	$I(u)$	$I(v_1)$	$I(v_2)$	$I(v_3)$	$I(v_4)$
0	u	0	0	0	0	0
1	v_2	1	0	0	0	0
2	v_3	1	0	1	0	0
3	v_4	1	0	1	1	0
4	v_1	1	0	1	1	1
5	v_2	1	1	1	1	1
6	v_1	1	1	2	1	1
7	u	1	2	2	1	1
8	v_4	2	2	2	1	1
9	v_3	2	2	2	1	2
10	u	2	2	2	2	2
11	—	3	2	2	2	2

从无向图的中国邮路问题可以推广到有向图的中国邮路问题，它的求解与最小费用流问题的求解是类似的，我们将在第九章中再作介绍。

下面我们举一个有向欧拉图的应用实例。

例 8.7 模数转换问题

计算机磁鼓设计就是一个模数转换问题。

一个旋转鼓轮，表面上分成 16 块扇形区域，每一块分别由导体或绝缘体组成，如图8.27（a）所示，图（b）是它的剖面图。图中阴影区表示导体，空白区表示绝缘体，鼓的位置信息用二进制数表示，通过鼓面上的 4 个触点将信息输出，

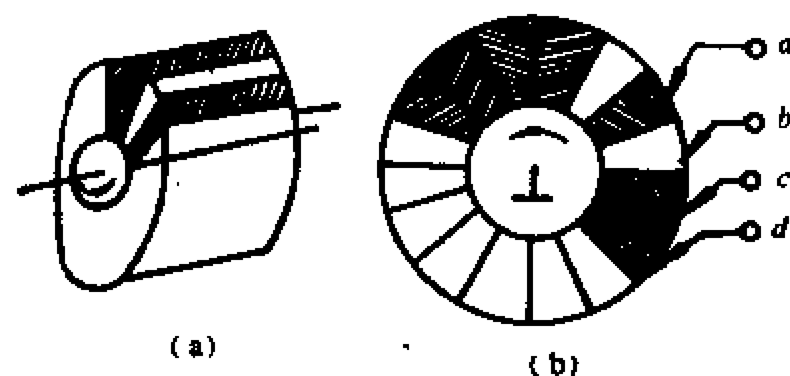


图 8.27

假定触点的接触面为导电区时，输出信息“0”，与绝缘区接触时输出信息“1”，如图(b)现在所处的位置，输出端 $abcd$ 输出的信息为“0100”，鼓逆时针方向旋转，则下一位置输出的信息为“1001”，因此每转一步输出一个四位二进制数码，16个扇形区，鼓轮旋转一周，将输出16个四位二进制数码。

现在提出一个问题：导电区和绝缘区应该怎样安排，才能使每一步输出的信息均不相同？或者说应该怎样将16个“0”或“1”的数码组成一个园形排列，使得任意四个邻接的数字组成的数码均不相同。(这样的数列称为布鲁英数列)。

满足上述要求的排列是能够做到的。

我们构造一个有 8 个结点的有向图D，它的结点分别用三位二进制数表示，即

000, 001, 010, 011, 100, 101, 110, 111

从每一结点引出两条弧，如图8.28所示，如果结点的编码是 $a_1a_2a_3$(a_i 为二进制数，即为 0 或 1)，它的一条引出弧编码为 $a_1a_2a_30$，且终止于编码是 a_2a_30的结点上，另一条引出弧编码为 $a_1a_2a_31$，且终止于编码为 a_2a_31 的结点上。

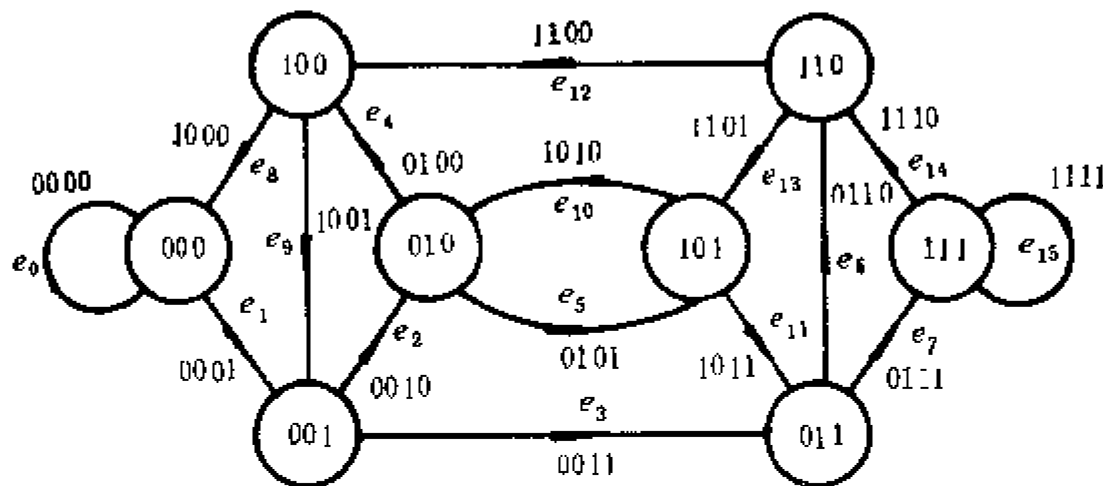


图 8.28

例如：从结点 001 引出两条弧，一条弧是0010，它终止于结点 010 上，另一条弧是0011，终止于结点 011 上，因此结点 000 和 111 引出的弧将有一条形成自环，它们的标号分别是 0000 和 1111。

于是从结点共引出 16 条弧(包括自环)，这16条弧由16个不同的四位二进制数码表示，在图的一条通路中，任意两条顺向连接的弧，它们的编码必然是 $a_1a_2a_3a_4 \rightarrow a_2a_3a_4a_5$，即前一条弧数码的后三位必然与后一条弧的前三位相同。而且这16条弧的编码是互不相同的。

可以看出，图D每一结点的引入次数都等于它的引出次数，所以D是有向欧拉图，从图的任一结点出发，沿弧的方向可走遍每条弧一次且仅一次又回到原点，例如从结点 000 出发可得到有向欧拉回路如下(用弧的序列表示)：

$$e_0\ e_1\ e_2\ e_5\ e_{10}\ e_4\ e_9\ e_3\ e_6\ e_{13}\ e_{11}\ e_7\ e_{15}\ e_{14}\ e_{12}\ e_8$$

对应这16条弧的四位二进制数的序列为：

0000101001101111

我们如果将鼓轮的扇形区按上面的序列进行安排，即对应“0”处是导电区，对应“1”处是绝缘区，则鼓轮旋转时，必将依次输出16个不同的四位二进制数码。

从上面的分析，我们可以推广到鼓轮具有 n 个触点的情况，即如果要求输出 n 位二进制数码，这时，鼓轮应分成 2^n 个扇形区，我们可以构造一个具有 2^{n-1} 个结点的有向图，

每个结点都用（$n-1$）位二进制数码表示，从结点$a_1a_2\cdots a_{n-1}$出发（a_i为0或1），引出两条弧，一条弧的编码为$a_1a_2\cdots a_{n-1}0$，终止于结点$a_2\cdots a_{n-1}0$上，一条弧编码为$a_1a_2\cdots a_{n-1}1$，终止于结点$a_2\cdots a_{n-1}1$上，显然这个图每个结点的引入次数都等于它的引出次数，图是一个有向欧拉图。

§8.5 H 图

所谓H图起源于一种游戏，1859年英国数学家哈密尔顿（Hamilton）提出一种名叫环游世界的游戏，即在地球上给定20个城市a，b，…，s，t，能否从一个城市出发经过每个城市一次且仅一次，最后回到原出发地。

当时哈密尔顿用一个正十二面体的20个顶点代表20个城市，这个正十二面体同构于一个平面图如图8.29所示。要求从一个顶点出发，沿着十二面体的棱经过每个顶点一次且仅一次最后回到原点。这个游戏曾经风靡一时，它有若干个解，图中顶点用数字序列表示的一条路径，就是其中的一个解。即

1，2，3，4，5，6，7，8，9，10，11，12，13，14，15，16，17，18，19，20，1。

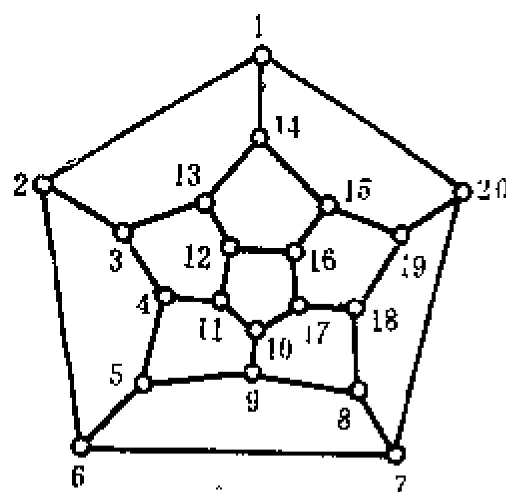


图 8.29

定义 8.6 一个无向图G，若存在一条通过所有结点一次且仅一次的路径，称为哈密尔顿路径。若存在通过所有结点一次且仅一次的回路，称为哈密尔顿回路。具有哈密尔顿回路的图称为哈密尔顿图，简称H图。

表面看来，哈密尔顿回路与欧拉回路很相似，但判断一个图是否是H图，要比判断一个图是否是E图困难得多。对于E图，已经有一个充分必要的判别定理，它是行之有效的。但要判别H图却远非如此简单，在这方面的探索一直在继续，下面介绍一些比较成熟的判别定理。

定理 8.10 若无向图$G=(V, E)$是H图，则对结点集合V的任一非空子集S，下式成立

$$W(G-S)\leqslant|S| \tag{8.1}$$

这里$W(G-S)$表示子图$(G-S)$的连通分支数。

证：设C是图G的一条H回路，S是V的任一非空子集。对S用归纳法证明。

如果S只有一个结点，设为v_1，将此结点从C中除去，显然$C-v_1$是一条连通的路径，子图（$G-S$）也是连通图，即$W(G-S)=|S|=1$，式（8.1）成立。

如果S有两个结点，设为v_1和v_2，则去掉v_1和v_2之后，$C-S$可能有两种情况：

（1）v_1与v_2是邻接的，则$C-S$仍然是一条连通的路径，子图（$G-S$）也是连通图，即$|S|=2$而$W(G-S)=1$，式（8.1）成立。

（2）v_1与v_2不邻接，去掉v_1和v_2之后C被分成两段路径，此时（$G-S$）可能仍然连通，最多被分成两个连通分支，即$W(G-S)\leqslant 2$，则式（8.1）亦成立。

设$|S_n|=n$时式（8.1）成立，当$|S_{n+1}|=n+1$时，若增加的结点v_{n+1}与原有的n个

结点 $v_1, v_2, \cdots, v_n$ 中的任一个结点邻接，则 $C-S_n$ 和 $C-S_{n+1}$ 被分成的段数相同，此时 $W(G-S_{n+1})=W(G-S_n)\leqslant n<n+1$，若 v_{n+1} 与 $v_1, v_2, \cdots, v_n$ 中的任一结点均不邻接，则 $W(G-S_{n+1})\leqslant W(G-S_n)+1\leqslant n+1$，式（8.1）亦成立。 ▮

定理 8.10 给出的是判定H图的必要条件而非充分条件，所以满足式（8.1）的图不一定是H图。例如图8.30所示的彼特森图，可以看出，去掉任意一个或两个结点，图仍然是连通的，去掉 3 个结点，图最多被分成两个连通分支，去掉 4 个结点，图最多被分成 3 个连通分支，去掉 5 个以上的结点，剩下的子图结点数将小于等于 5，因此不可能有 5 个以上的连通分支，就是说对任意非空子集 S，式（8.1）都是成立的，但彼特森图却是一个典型的非H图。

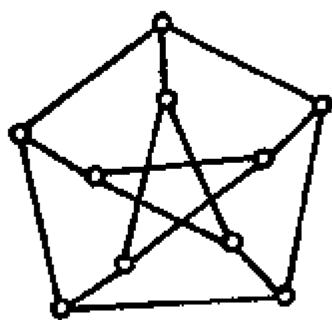


图 8.30

当然，如果不满足式（8.1）条件的图，一定不是H图，因此用这一定理判定非H图，有时更为有效。

定理 8.11 任意一个完全图 K_n 是H图。

结论是显然的，但定理提出的是一个充分条件而非必要条件，例如图 8.29 所示的图是H图但并不是完全图。

定理 8.12 如果图 $G=(V, E)$ 满足条件

$$\delta\geqslant\frac{1}{2}n \tag{8.2}$$

则G是H图。

式中 $n=|V|\geqslant 3$，δ 为结点的最小次数。

证：先证满足式（8.2）的图一定是连通图。用反证法，设G不是连通图，则至少有两个连通分支，设一个分支的结点子集为 V_1，则另一分支的结点子集为 $V-V_1$，设 $|V_1|=k$，则 $|V-V_1|=n-k$，对任一结点 $V_i\in V_1$，必有

$$deg(v_i)\leqslant k-1$$

同理，对任一 $v_j\in V-V_1$，必有

$$deg(v_j)\leqslant n-k-1$$

如果 $deg\,(v_i)\geqslant\frac{1}{2}n$，即 $k-1\geqslant\frac{1}{2}n$，则 $n-k-1\leqslant\frac{n}{2}-2<\frac{n}{2}$，即 $deg\,(v_j)<\frac{n}{2}$。同理，如果 $deg(v_j)\geqslant\frac{n}{2}$，则 $k-1\leqslant\frac{n}{2}-2<\frac{n}{2}$，即 $deg(v_i)<\frac{n}{2}$。与题设的条件矛盾，若图是有两个以上分支的非连通图，上述矛盾仍然存在，所以G必然是连通图。

下面证明G是H图。

仍用反证法，设上述条件满足，但G不是H图，由定理 8.11 可知G一定不是完全图。因此可以在图G中不断添加边而不致破坏原有条件。设边添加到某种程度时已构成一条从 v_1 到 v_n 的哈密尔顿路径，并设此时的图为 G^*，显然如果在 G^*上再添加一条边 (v_1, v_n)，就会成为H图，即 $G^*+(v_1, v_n)$ 是H图。

回到图 G^*，设此时的哈密尔顿路径为：

$$v_1, v_2, \cdots, v_i, v_{i+1}, \cdots, v_{n-1}, v_n$$

如图 8.31 所示

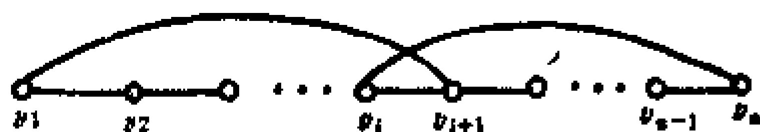


图 8.31

我们定义两个结点子集如下：

$$V_1=\{v_i\mid(v_1, v_{i+1})\in E\}$$

$$V_2=\{v_j\mid(v_j, v_n)\in E\}$$

显然有 $deg(v_1)=|V_1|$ 和 $deg(v_n)=|V_2|$，而且 $V_n\notin V_1$ 及 $v_n\notin V_2$。

其次，$V_1\cap V_2=\phi$，即 V_1 与 V_2 无公共结点，否则如有公共结点，设为 v_i，则有 $(v_1, v_{i+1})\in E$ 且 $(v_i, v_n)\in E$，这时将形成回路（见图8.31）：

$$v_1, v_2, \cdots, v_i, v_n, v_{n-1}, \cdots v_{i+1}, v_1$$

与题设矛盾，因此证明了 V_1 与 V_2 无公共结点。

由于 V_1 和 V_2 都是 V 的子集，它们又无公共结点，且都不包含结点 v_n，故必有

$$|V_1|+|V_2|<|V|=n$$

即
$$deg(v_1)+deg(v_n)<n$$

因此必有 $deg(v_1)<\frac{1}{2}n$ 或 $deg(v_n)<\frac{1}{2}n$ 两者之一成立，与式（8.2）所给的条件矛盾，所以 G 是 H 图。 ▮

定理 8.13 设 G 是一个有 n 个结点的简单无向图，若 G 的任意两个结点 v_i，v_j，均有

$$deg(v_i)+deg(v_j)\geqslant n-1 \qquad (8.3)$$

则 G 中存在一条哈密尔顿路径

证：首先证明 G 是连通图。用反证法，设 G 不是连通图，则至少有两个连通分支，设一个分支的结点数为 n_1，另一分支的结点数为 n_2，应有 $n_1+n_2=n$，在结点数为 n_1 的分支中，结点的最大次数为 n_1-1，同理，在另一分支中结点的最大次数为 n_2-1，这是两个次数最大的结点，它们的次数和为 $(n_1-1)+(n_2-1)=n-2<n-1$，与题设矛盾。若图是两个以上的连通分支构成的非连通图，这一矛盾仍然存在，故 G 不可能是非连通图。

其次证明 G 存在一条哈密尔顿路径，用构造法证明如下。

在图 G 中任意找一条有 m 个结点的基本路径 $P=v_1, v_2, \cdots, v_m$，$(m\geqslant 2)$。若 $m=n$，则这条基本路径通过全部结点，因而它就是哈密尔顿路径，命题得证。

若 $m<n$，则考察路径的两个端点 v_1 与 v_m 是否还与路径之外的某一结点比如 v_x 邻接，如果邻接，就把路径延伸到 v_x，使路径为 $v_x, v_1, v_2, \cdots, v_m$ 或 $v_1, v_2, \cdots, v_m, v_x$，按此方法继续进行下去直到路径的两个端点不再与路径外的任一结点邻接，以致路径不能再延伸为止，为了表示方便，我们把不能再延伸的路径仍记作

$$P=v_1, v_2, \cdots, v_m$$

现在我们要证明：存在一条回路，它通过 P 的所有结点 $v_1, v_2, \cdots, v_m$。

首先考察端点 v_1，假设它和路径上的 k 个结点邻接，这 k 个结点记为 v_{i_1}(即 v_2)，$v_{i_2}, \cdots, v_{ik}$，这里 v_{ij} $(1\leqslant j\leqslant k)$ 是路径 P 上的结点，于是 v_1 的次数为

$$deg(v_1)=k$$

在路径 P 上按顺序将在前面与 $v_{i_1}, v_{i_2}, \cdots, v_{ij}$ 邻接的结点记作 v_{i_1-1}，$v_{i_2-1}, \cdots, v_{ij-1}$，

如图8.32（a）所示，可以肯定，另一端点v_m至少与这k个结点中的一个邻接，否则路径上与v_m邻接的结点最多有（$m-1-k$）个（因为路径共有k个结点），即$deg(v_m)\leqslant(m-1-k)$，则$deg(v_1)+deg(v_m)\leqslant m-1<n-1$，与题设矛盾。因此$v_m$必与$v_{i_1-1}$，$v_{i_2-1}$，…，$v_{i_j-1}$中至少一个结点邻接，不妨设这一结点为$v_{i_j-1}$，如图（b）所示，显然存在一回路：$v_1$，$v_2$，…，$v_{i_j-1}$，$v_m$，$v_{m-1}$，…，$v_{i_j}$，$v_1$

至此，在图G中我们得到了一条含有m个结点的基本回路，在这个回路之外尚有$n-m$个结点，由于G是连通图，回路外的$n-m$个结点中，至少有一个与回路上的某一结点邻接，不妨设回路外的结点v_x与回路上的结点v_t邻接，如图8.32(b)所示，于是就可得

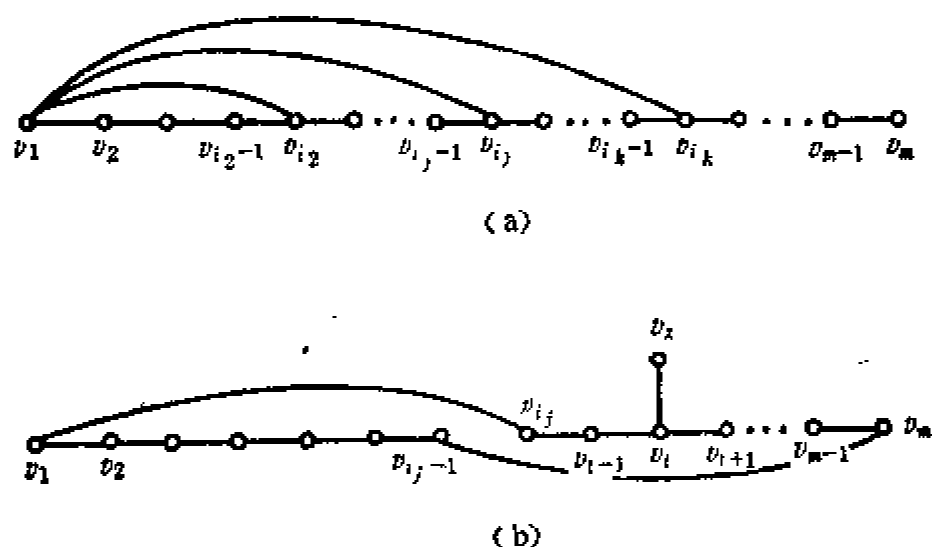


图 8.32

到一条以v_x和v_{t+1}为端点的基本路径P'如下：

$$P'=v_x,\ v_t,\ v_{t-1},\ v_{ij},\ v_{\cdot},\cdots,\ v_{ij-1},\ v_m,\ v_{m-1}\cdots v_{t+1}$$

显然，路径P'的结点数比原来P的结点数多1，即路径又延伸了一个结点。

这时又可重复前面的过程，即若P'的两个端点v_x和v_{t+1}与P'外的结点邻接，又可将P'延伸，直到两个端点不再与路径外的结点邻接为止，于是又得到一个包含更多结点的回路，通过回路上的结点与回路外的结点之间的关联边，又可得到一条比回路结点数多1的基本路径。如此继续下去，由于图的有限性，最终得到的基本路径必通过图的所有结点，它就是哈密尔顿路径。 ▌

定理 8.14 设G是一个有n个结点的简单无向图，若G的任意两个结点v_i，v_j，均有

$$deg(v_i)+deg(v_j)\geqslant n \tag{8.4}$$

则G是H图。

证：由定理8.13已知，G有一条哈密尔顿路径v_1，v_2，…，v_n，若v_1与v_n邻接，则构成H回路，命题得证。否则设v_1与v_{i_1}，v_{i_2}，…，v_{ik}共k个结点邻接，那么v_n至少与v_{i_1-1}，v_{i_2-1}，…，v_{ik-1}中一个结点邻接，不妨设与v_{ij-1}邻接，如图8.33所示，则存在回路

$$v_1,\ v_2,\cdots v_{ij-1},\ v_n,\ v_{n-1},\ \cdots,\ v_{ij},\ v_1$$

这是一条H回路，故G是H图。 ▌

上面三个定理给出的都是判别H图的充分条件，而非必要条件，例如图8.34并不具备定理提出的条件，但它却是H图。

定义 8.7 若图$G=(V,E)$，$|V|=n$，若存在结点v_i，v_j满足$(v_i,\ v_j)\notin E$且$deg(v_i)+deg(v_j)\geqslant n$，则将边（$v_i$，$v_j$）加到$G$上，如此反复进行直到不能再添加边为止，此时得

到的图称为G的闭包，记作G^+。

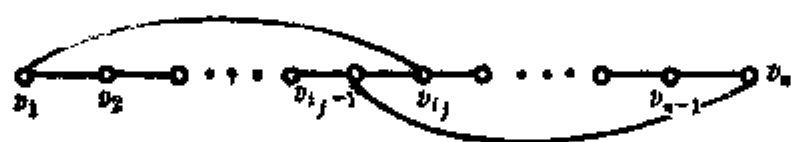


图 8.33

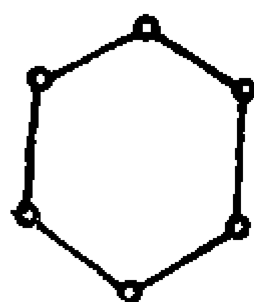


图 8.34

图8.35表明了图G的闭包构造过程，这里得到的闭包是一个完全图。但应指出图的闭包不一定是完全图，例如图 8.36 的图G，它的闭包 G^+就不是完全图。

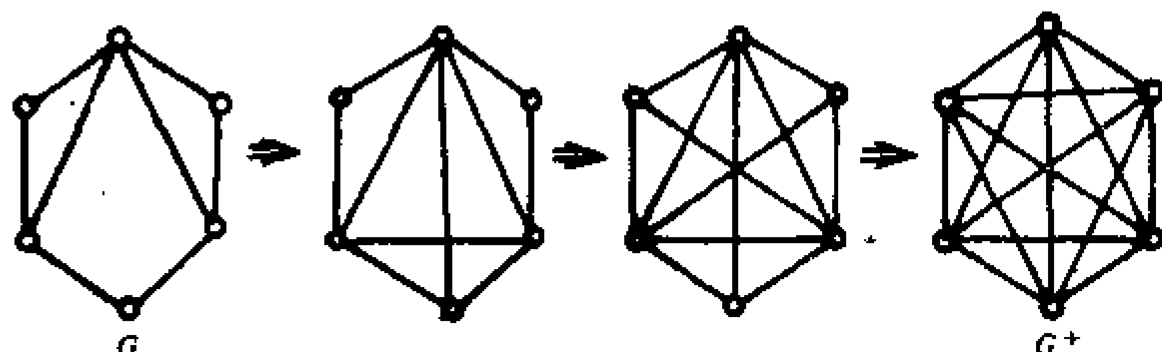


图 8.35

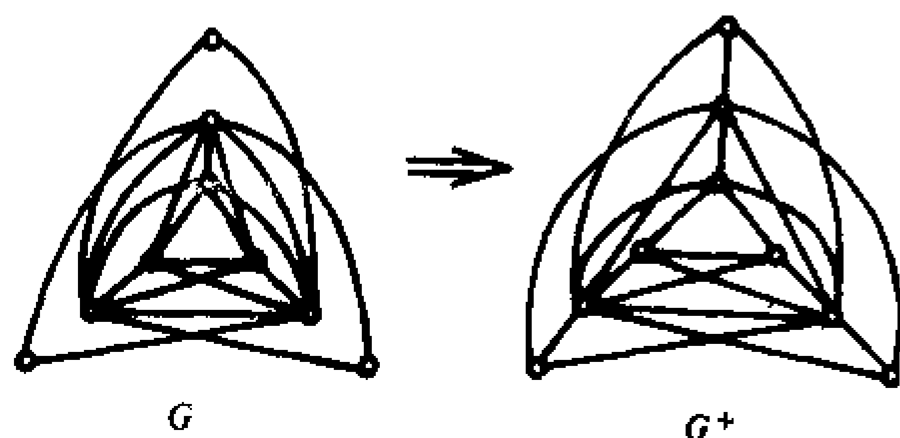


图 8.36

定理 8.15 图G的闭包 G^+是唯一的。

证：用反证法，设 G_1^+和 G_2^+是G的两个不同的闭包，G_1^+是按定义 8.7 的构造方法在 G 上增加边 e_1，e_2，…，e_p 得到的，即

$$G_1^+=G+\{e_1,e_2,\cdots,e_p\}$$

G_2^+是按同样方法在G上增加边 f_1，f_2，…，f_q 得到的，即

$$G_2^+=G+\{f_1,f_2,\cdots,f_q\}$$

不失一般性，设 G_1^+和 G_2^+的增加边中有 k 条是相同的，即 $e_i=f_i(0\leqslant i\leqslant k)$，而边 e_{k+1}，e_{k+2}，…不是 G_2^+的增加边、令

$$H=G+\{e_1,\ e_2,\cdots,e_k\}$$

则H既是 G_1^+的子图也是 G_2^+的子图，由子 e_{k+1} 是 G_1^+的增加边，设 $e_{k+1}=(u,\ v)$，则应有

$$deg(u)+deg(v)\geqslant n$$

上述关系在H中成立，按构造法则边 $e_{k+1}=(u,\ v)$ 应增加到 G_2^+上，即 e_{k+1} 也是 G_2^+的增加边，依此类推，G_1^+中的任一条增加边也是 G_2^+的增加边，反之亦然，因此 $G_1^+=G_2^+$。 ▮

定理 8.16 设G是n阶简单图，u和v是G中两个不邻接的结点，若

$$deg(u)+deg(v)\geqslant n$$

则G是H图的充要条件是$G+(u,\ v)$是H图。

证：若G是H图，即G存在H回路，增加任何一条边都不会破坏H回路的存在，所以$G+(u,\ v)$必然也是H图。

反之，若$G+(u,\ v)$是H图，则存在H回路，如果边$(u,\ v)$不在H回路中，则去掉$(u,\ v)$不会破坏H回路的存在，故$G=(G+(u,\ v))-(u,\ v)$仍含H回路，即G也是H图。

若$(u,\ v)$是图$G+(u,\ v)$的H回路中的一条边，不妨设H回路为

$$C=u,\ v,\ v_3,\ v_4,\cdots,\ v_n,\ u$$

如图8.37所示。此时图的全部n个结点都在回路C上，设结点$v_3,\ v_4,\cdots,\ v_{n-1}$中有$k$个结点与$u$邻接，不妨设这$k$个结点为$v_{i_1},\ v_{i_2},\cdots v_{i_k}$，则

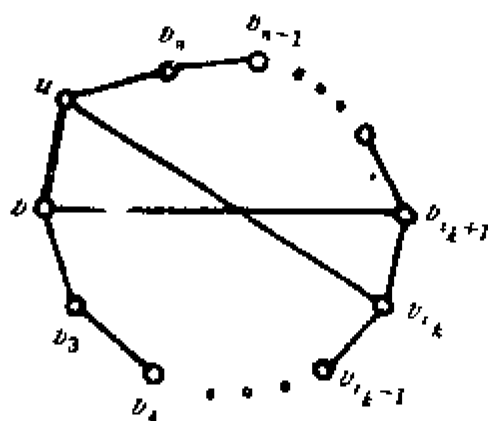


图 8.37

$$deg(u)=k+2$$

与定理8.13的证明类似，可以肯定，在结点$v_{i_1+1},v_{i_2+1},\cdots,\ v_{i_k+1}$ k个结点中，至少有一个结点与v邻接，不然的话与v邻接的结点最多有$(n-1-k)$个，即

$$deg(v)\leqslant n-1-k$$

则
$$deg(u)+deg(v)\leqslant n+1 \qquad (A)$$

根据定理给的条件，在图G中，u和v的次数和不小于n，因此在图$G+(u,\ v)$中，u和v的次数和不小于$n+2$，即

$$deg(u)+deg(v)\geqslant n+2 \qquad (B)$$

可见(A)式与(B)式矛盾，这就证明在$v_{i_1+1},\ v_{i_2+1},\cdots,\ v_{i_k+1}$中至少有一个结点与$v$邻接，设这个结点是$v_{i_k+1}$，如图8.37所示，则去掉$(u,\ v)$后，仍然存在回路

$$C'=u,\ v_{i_k},\ v_{i_k-1},\cdots,\ v_3,\ v,\ v_{i_k+1},\cdots v_n,\ u$$

显然C'也是H回路，故G是H图。 ▌

定理 8.17 一个简单图G是H图当且仅当它的闭包G^+是H图

证：设G的闭包是在G上添加m条边得到的，即

$$G^+=G+e_1+e_2+\cdots+e_m$$

若G是H图，则添加任何多条边都不会破坏H回路的存在，故G^+也是H图。

反之，若G^+是H图，根据定理8.16，每次删除一条添加的边得到的图仍是H图，如此反复进行，直到删除添加的m条边，因而图G也是H图。 ▌

定理 8.18 对n阶完全图K_n，若n为不小于3的奇数，则K_n有$(n-1)/2$个边不相交的H回路。

证：由定理8.11知K_n具有H回路，现将图的n个结点作如下排列，即将v_1置于一个圆的圆心上，其余结点则均匀地分布在圆周上，如图8.38所示。这时可得到图的一个H回路。

$$C_1=v_1\ v_2\ v_3\ v_4,\cdots,v_{n-2}\ v_{n-1}\ v_n\ v_1$$

现在将圆周按顺时针方向旋转，每次转动的角度为$\alpha=360°/(n-1)$，则每转一个α

角时，圆周上的结点恰好顺时针转到下一个结点的位置。例如图 8.38，当转第一个 α 角时，结点 v_4 转到 v_2 的位置，v_2 转到 v_3 的位置，v_{n-2} 则转到 v_n 的位置，如此等等，这时将得到另一 H 回路

$$C_2 = v_1v_4v_2v_6v_3\cdots, v_{n-1}v_{n-4}v_nv_{n-2}v_1$$

显然 C_1 与 C_2 没有公共边。圆周旋转一周，共旋转（$n-1$）次，得到（$n-1$）个 H 回路，但是上半周的旋转与对应的下半周的旋转，得到的 H 回路是相同的，例如旋转 180°角时，H 回路为

$$C_1' = v_1v_nv_{n-1}v_{n-2}\cdots v_4v_3v_2v_1$$

即 $C_1' = C_1$，所以在（$n-1$）个 H 回路中，边不相交的 H 回路只有 $(n-1)/2$ 个。 ▮

这一定理可以用来解决所谓圆桌排座问题，举例如下

例 8.8 有 11 个成员每天中午在同一张圆桌上就餐，希望安排每次就餐时邻座的人均不相同，问这种安排最多能持续多少天？

解：这个问题可化为求完全图 K_{11} 边不相交的 H 回路的问题。根据定理 8.18 的证明可知，存在 $(11-1)/2=5$ 个不相交的 H 回路，故这种安排最多可以持续 5 天。图 8.39 说明了这种安排方法。

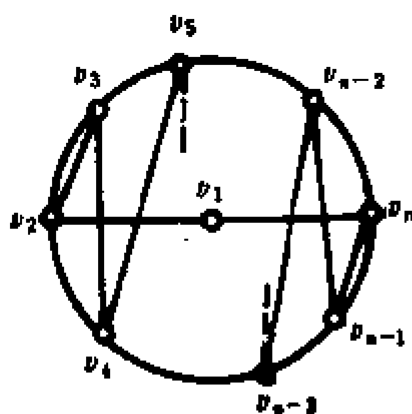


图 8.38

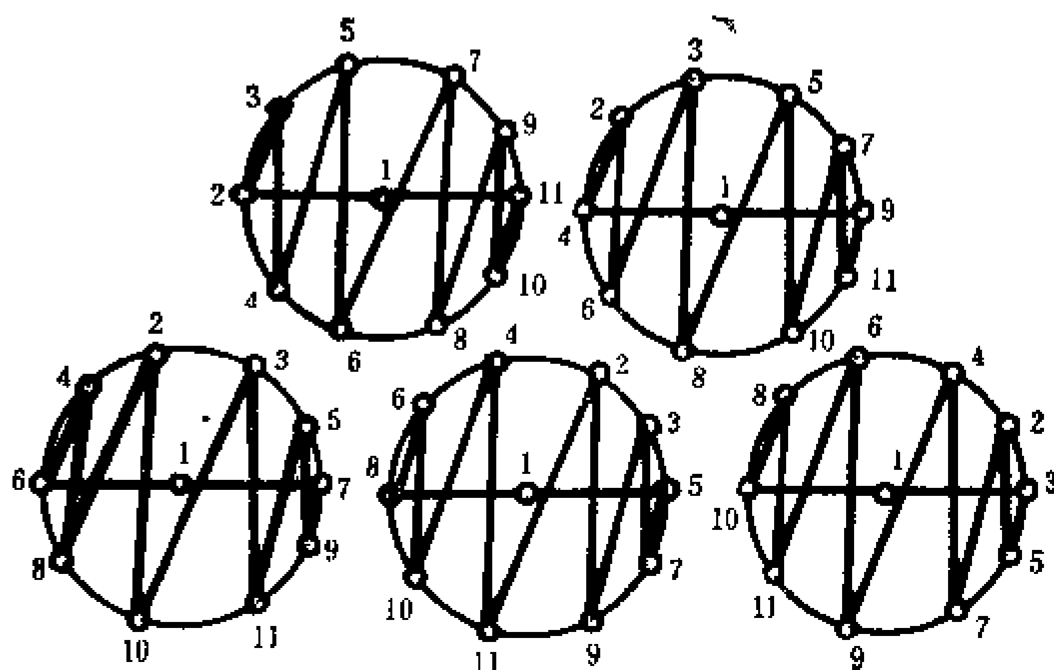


图 8.39

由图可见这 5 次不同座位的安排为：

1	2	3	4	5	6	7	8	9	10	11	1
1	4	2	6	3	8	5	10	7	11	9	1
1	6	4	8	2	10	3	11	5	9	7	1
1	8	6	10	4	11	2	9	3	7	5	1
1	10	8	11	6	9	4	7	2	5	3	1

定理8.18是针对结点数目为奇数的图而言的，对于一般图，边不相交的 H 回路的数目尚未得到解决。

定理 8.19 在 n 阶完全图 K_n 中，不同的 H 回路（含边相交的和边不相交的）共有 $(n-1)!/2$ 个。

证：完全图中任意两结点都是邻接的，所以从第一个结点到第二个结点的路径可以有

$(n-1)$ 种选择，从第二个结点到第三个结点的路径又可以有 $(n-2)$ 种选择，依此类推，最后从第 $(n-1)$ 个结点到第 n 个结点的路径只有一种选择，因此共有 $(n-1)!$ 种选择构成 H 回路，但上面的每种选择中都重复了一次，例如回路 $v_1v_2v_3\cdots v_{n-2}v_{n-1}v_nv_1$ 与回路 $v_1v_nv_{n-1}v_{n-2}\cdots v_3v_2v_1$ 实际是同一 H 回路。所以不同的 H 回路的数目应是 $(n-1)!$ 的一半，即 $(n-1)!/2$ 个。 ▌

§8.6 有向 H 图

定义 8.8 在有向图 D 中，经过每个结点一次的有向路径称为有向 H 路径。经过每个结点一次的有向回路，称为有向 H 回路。具有有向 H 回路的有向图，称为有向 H 图。

定理 8.20 若有向图 D 的底图是完全图，则 D 存在一条有向 H 路径。

证： 设有向图 $D=\langle V, E\rangle$ 的底图是一个 n 阶完全图，$n=|V|$，因此对任意两结点 $v_i, v_j\in V$，如 $\langle v_i, v_j\rangle\notin E$，必有 $\langle v_j, v_i\rangle\in E$。

在有向图 D 中，先任意找一条有 k 个结点的有向基本路径 P：$v_1, v_2, \cdots, v_k$。

若 $k=n$，则 P 就是有向 H 路径，否则说明 P 还没有经过 D 的所有结点，可以采用插入结点法扩展这条路径，步骤如下：

对任意一个不在路径上的结点 v_x，考察路径 P 的起点 v_1 与 v_x 的邻接关系，如 $\langle v_x, v_1\rangle\in E$，则可将 v_x 加到路径的左端，路径成为

$$v_x, v_1, v_2, \cdots, v_k$$

此时路径扩展了一个结点。若 $\langle v_x, v_1\rangle\notin E$，则必有 $\langle v_1, v_x\rangle\in E$，这时可考察 v_x 与 v_2 的邻接关系。如 $\langle v_x, v_2\rangle\in E$，可将 v_x 插入 v_1 和 v_2 之间，路径成为

$$v_1, v_x, v_2, \cdots, v_k$$

此时路径亦扩展了一个结点，如 $\langle v_x, v_2\rangle\notin E$，则可考察 v_x 与 v_3 的邻接关系，如此继续下去，若 v_x 不能插入 $v_1, v_2, \cdots, v_k$ 之间，则必有 $\langle v_k, v_x\rangle\in E$，可将 v_x 加到路径的尾端而成为

$$v_1, v_2, \cdots, v_k, v_x$$

因此，对路径外的任一结点，利用上述法则一定可以插入到路径中，使原来经过 k 个结点的路径扩展到经过 $k+1$ 个结点。于是可以不断地将路径外的结点加到路径上，由于图的有限性，总可以通过有限步骤将图的全部结点加到路径上，最后得到的即是有向 H 路径。 ▌

由定理的证明过程可知，最初的有向路径 P 可以从只含一条弧 $\langle v_i, v_j\rangle$ 开始，不断地加入结点而扩展成为有向 H 路径。

例 8.9 如图 8.40 所示的有向图，它的底图是完全无向图，因此存在一条有向 H 路径，可以看出这条路径是：v_4, v_1, v_5, v_3, v_2。

定理 8.21 若强连通有向图 D 的底图是完全图，则 D 是有向 H 图。

证： 设 $D=\langle V, E\rangle$ 是满足所给条件的有向图，由于 D 是强连通图，一定存在一个有向基本回路，因此一定存在一个长度最长的基本回路，设 $C=(v_1, v_2, \cdots v_m, v_1)$ 是 D 中最长的一个有向基本回路，下面我们要证明 C 就是有向 H 回路。

采用反证法，设 C 不是有向 H 回路，即还有一些结点不在回路 C 上，设不在 C 上的结

点为 v，那么对 C 上的任一结点 v_i

（1）若 $\langle v, v_i\rangle \in E$（如图 8.41 所示）则必有 $\langle v_{i-1}, v\rangle \notin E$。否则可将 v 插入回路的

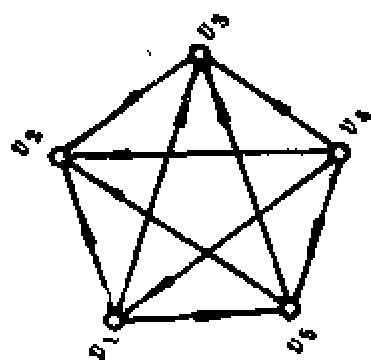


图 8.40

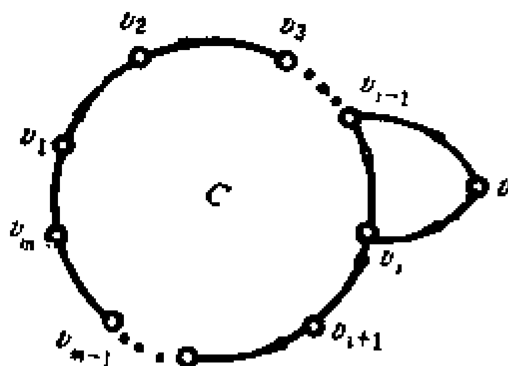


图 8.41

v_{i-1} 和 v_i 之间而使回路的长度增加，这与 C 是最长的回路矛盾。另一方面，因 D 的底图是完全图，若 $\langle v_{i-1}, v\rangle \notin E$，则必有 $\langle v, v_{i-1}\rangle \in E$，综合上述，即得

若 $\langle v, v_i\rangle \in E$，必有 $\langle v, v_{i-1}\rangle \in E$

因是完全图，v 与回路 C 上的每一结点，都有一条有向弧关联，由于 v_i 的任意性，可见这些有向弧都是由 v 引出而终止到 C 的各个结点上的。

（2）同理，若 $\langle v_i, v\rangle \in E$（如图 8.42 所示）则必有 $\langle v, v_{i+1}\rangle \notin E$，因而有 $\langle v_{i+1}, v\rangle \in E$，这时回路 C 上的各个结点，都将有一引出弧终止于结点 v 上。

因此，回路 C 外的结点可以分为两类，一类为属于第（1）种情况的结点，弧由这些结点引出而终止于 C 的各结点上，这一类结点集合记作 V'，另一类为属于第（2）种情况的结点，弧由 C 上的各结点引出而终止于这类结点上，这一类结点集合记作 V''，根据假设，有 $V' \cup V'' \neq \phi$，如图 8.43 所示，而且有 $V' \neq \phi$ 及 $V'' \neq \phi$，这是因为如果只存在 V'，则回路 C 上的点不能到达 V' 中的点，与强连通的假设矛盾，并且必然存在有向弧从 V'' 中的结点引向 V' 中的结点，设其中一条为 $\langle v'', v'\rangle$ 如图所示，于是我们将得到回路

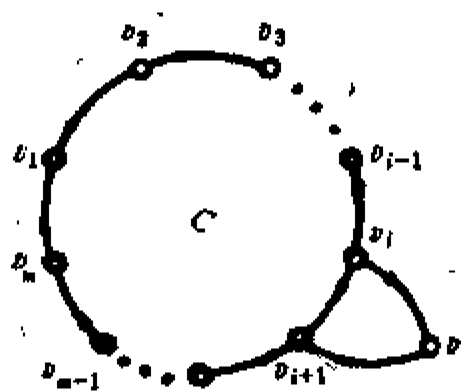


图 8.42

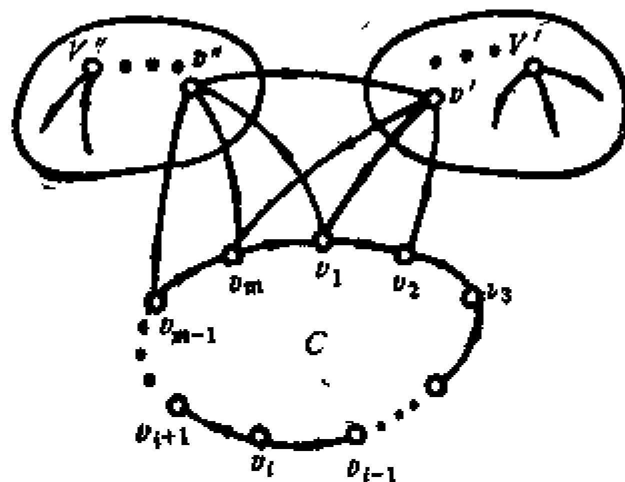


图 8.43

$$C' = (v_1, v_2, \cdots, v_m, v'', v', v_1)$$

可见 C' 的长度大于 C，与 C 是最长回路的假设矛盾，所以 C 是 H 回路。 ▌

在上述定理的证明中，任意两结点之间，我们只取一条单向弧（但满足强连通条件），事实上，在某些强连通有向图中，可能存在结点对 v_i 和 v_j，既有 $\langle v_i, v_j\rangle \in E$，也有 $\langle v_j, v_i\rangle \in E$，从定理的证明过程可知，如果出现这种情况，并不影响我们证明的方法，也不会影响由此而得出的正确结论。

例 8.10 如图 8.44 所示的强连通有向图，它的底图是完全图，因此该图是有向 H

图，H回路为

$$v_1,\ v_2,\ v_3,\ v_5,\ v_4,\ v_1$$

定理8.21给出的条件也是充分条件而非必要条件。但是，若有向图D不是强连通的，只是底图是完全图，则D不一定是H图（但一定具有H路径），另一方面，若D是强连通的但底图不是完全图，则D不一定是H图，甚至也不含H回路，图 8.45 所示的有向图就是一个例子。

定理 8.22 设D是具有n个结点的简单强连通有向图，若对D的任一结点v，均有

$$deg^+(v)+deg^-(v)\geqslant n \tag{8.5}$$

则D是有向H图。

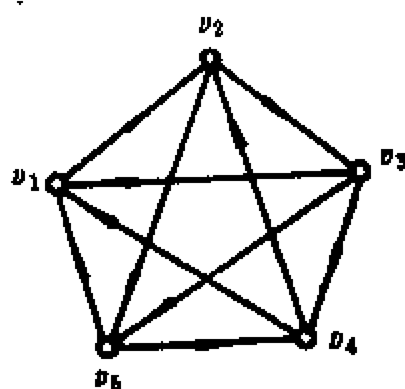


图 8.44

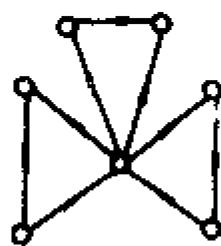


图 8.45

定理由 Ghouila—Houri 于 1960 年提出，证明十分冗长，此处从略，有兴趣的读者可参阅〔5〕。

§8.7 H图的寻迹

到目前为止，还没有找到H图存在的充分必要条件，因此，任给一个图，判定它是否含有H回路（路径）仍然是一个难题。在寻迹H图时，人们也提出了多种算法，下面介绍的几种算法，在图的规模不是很大时，还是比较好的。

一、标记法

用此法可以判定一个无向图是否存在H路径。

方法为：在图中任意指定一个结点，并给这一结点标号为A，然后把凡是与结点A邻接的结点，都标号为B，又把凡与B邻接的结点标为A，如此继续下去，直到所有的结点

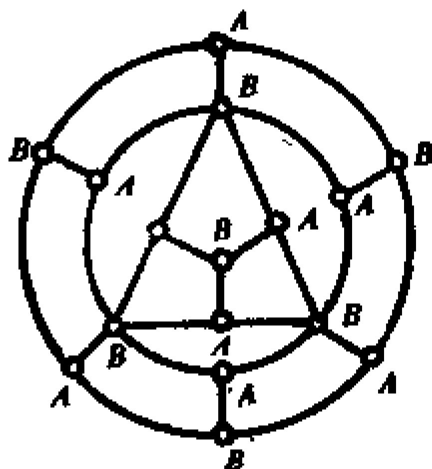


图 8.46

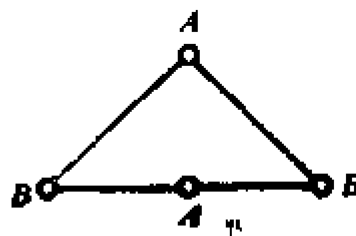


图 8.47

都标号完毕。如果存在H路径，则标号为A，B的结点应交替出现，A，B结点数目之差不应大于1。

如图8.46，按上述方法标号后，标号A的结点有9个，标号B的结点有7个，因此该图不存在H路径。

在标号过程中，如果遇到相邻两结点出现相同标号时，可在此对应边上增加一个结点，并标上相异符号，如图8.47所示。

二、回溯法

回溯法的基本方法是搜索，一般是在给定条件下应用深度优先搜索法构成解的树形结构以便搜索容易实现。这一算法可以找出图的所有H回路。

设$G=(V, E)$是一个有n个结点的连通图，用向量$(x_1, x_2, \cdots, x_n)$表示回溯法的一组解，其中x_i表示在一个可能的H回路中第i次访问的结点号数，设已得到$(x_1, x_2, \cdots, x_{k-1})$，下一步就是怎样从未访问过的结点中找出$x_k$、算法中先输出图的邻接矩阵$GRAPH(1:n, 1:n)$，输出以$X(1)$，$X(2)$，…，$X(n)$表示$H$回路，为了避免重复，对结点编号后规定结点1为起始点，即$X(1)=1$，算法中回溯过程用HAMILTONIAN (k)，其中要调用寻找下一结点的过程NEXTVERTEX(k)，并执行赋值语句：$X(1)\leftarrow 1$，$X(2:n)\leftarrow 1$，然后执行语句调用HAMILTONIAN(2)。

```
HAMILTONIAN(k) （求所有H回路）
1. While X(k)≤n do
   begin
2. NEXTVERTEX(k) （找X(k)的下一结点）
3. if X(k)=0 then return （结束本次调用）
4. if k=n then （找到一个H回路）
5. Print (X,'1') （输出回路）
6. else HAMILTONIAN (k+1) （找下一个点）
   end
NEXTVERTEX (k) （产生下一个结点）
   begin
1. x(k)←(X(k)+1) mod(n+1) （找下一结点）
2. if X(k)=0 then return （找不到合适结点，结束本次调用）
3. if GRAPH (X(k-1), X(k))=1 then
   begin ((X(k-1), X(k))是图的一条边)
4. for j=1 to k-1 do
5. if X(j)=X(k) then goto 1
6. if k<n 或 k=n 且 GRAPH (X(n), 1)=1
   then return
   end
7. else go to 1
   end
```

例 8.11 如图8.48的无向图，用上述算法找出图的唯一一个H回路，回溯过程如图8.49所示的回溯树，其中结点的编号反映了过程的执行路线，从树根到结点A的路径正好构成一H回路，它是 1，2，8，7，6，5，4，3。而从树根到A'的H回路实际上与A是同一H回路，只是寻迹路线相反。

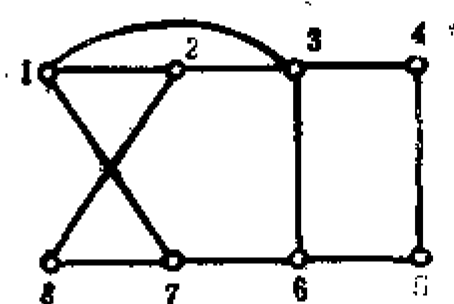


图 8.48

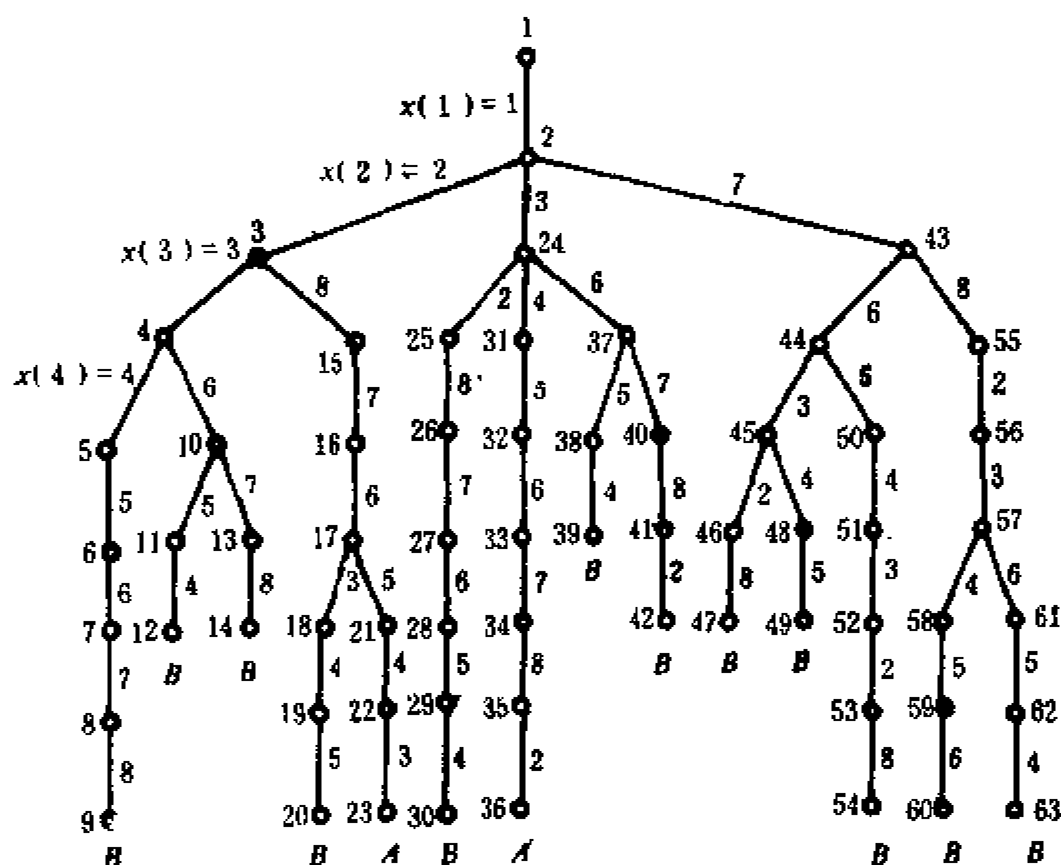


图 8.49

三、矩阵法

当图不是完全图时，可以应用这里介绍的矩阵算法求出图的所有H路径或H回路（如果存在的话），算法对有向图或无向图均适用，因此应用是较广的。

算法步骤如下：

1．给出图的邻接矩阵M_1，其中$M_1(i, j)$用边v_iv_j表示，若无此边则置0

2．求出矩阵M，它由M_1中删除每一个非零项的第一个结点而得。

3．定义矩阵的串乘法如下：

$$M_k = M_{k-1} * M$$

其中

$$M_k(i, j) = \left\{ \bigvee_t M_{k-1}(i, t) \circ M(t, j) \right\}$$

这里：

（1）“$\circ$”是特种乘法，其运算规则为

1° 零乘任何项均为零

2° 不为零的两项相乘是将此两项串联起来。

3° 两项中有公共元素时相乘为零。

（2）“$\vee$”是特种加法，加法的结果是项的并列。

4．$M_k(i, j)$表示从v_i到v_j长度为k的基本路径的集合，故$M_{n-1}(i, j)$即表示从v_i到v_j的H路径的集合（n为图的结点数）

现举一例说明如下。

例 8.12 试用矩阵乘法求图 8.50 所示的有向图的 H 回路。

解： 根据上述算法求出 M_1, M, M_2, M_3, M_4, M_5 如下：

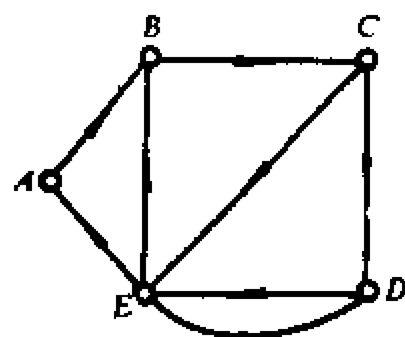


图 8.50

$$
\begin{array}{c c} & \begin{array}{ccccc} A & B & C & D & E \end{array} \\
M_1 = \begin{array}{c} A \\ B \\ C \\ D \\ E \end{array} & \begin{bmatrix} 0 & AB & 0 & 0 & 0 \\ 0 & 0 & BC & 0 & 0 \\ 0 & 0 & 0 & CD & CE \\ 0 & 0 & 0 & 0 & DE \\ EA & EB & 0 & ED & 0 \end{bmatrix}
\end{array}
$$

$$
M = \begin{bmatrix} 0 & B & 0 & 0 & 0 \\ 0 & 0 & C & 0 & 0 \\ 0 & 0 & 0 & D & E \\ 0 & 0 & 0 & 0 & E \\ A & B & 0 & D & 0 \end{bmatrix}
$$

$$
M_2 = \begin{bmatrix} 0 & 0 & ABC & 0 & 0 \\ 0 & 0 & 0 & BCD & BCE \\ CEA & CEB & 0 & CED & CDE \\ DEA & DEB & 0 & 0 & 0 \\ 0 & EAB & EBC & 0 & 0 \end{bmatrix}
$$

$$
M_3 = \begin{bmatrix} 0 & 0 & 0 & ABCD & ABCE \\ BCEA & 0 & 0 & BCED & BCDE \\ CDEA & \begin{pmatrix} CEAB \\ CDEA \end{pmatrix} & 0 & 0 & 0 \\ 0 & DEAB & DEBC & 0 & 0 \\ 0 & 0 & EABC & EABD & 0 \end{bmatrix}
$$

$$
M_4 = \begin{bmatrix} 0 & 0 & 0 & ABCED & ABCDE \\ BCDEA & 0 & 0 & 0 & 0 \\ 0 & CDEAB & 0 & 0 & 0 \\ 0 & 0 & DEABC & 0 & 0 \\ 0 & 0 & 0 & EABCD & 0 \end{bmatrix}
$$

$$
M_5 = \begin{bmatrix} ABCDEA & 0 & 0 & 0 & 0 \\ 0 & BCDEAB & 0 & 0 & 0 \\ 0 & 0 & CDEABC & 0 & 0 \\ 0 & 0 & 0 & DEABCD & 0 \\ 0 & 0 & 0 & 0 & EABCDE \end{bmatrix}
$$

说明：

（1）M_1 是图的邻接矩的特殊表示法，矩阵中的元素是用对应结点之间的弧表示的。

（2）M是进行矩阵乘法的矩阵单元，它是将M_1中非零元素的第一个结点去掉而形成的。

（3）$M_k=M_{k-1}*M$，例如$M_2(3,2)=CEB$，它是由M_1的第3行与M的第2列对应元素串乘而得，其中只有$M_1(3,5)=CE$及$M(5,2)=B$，故$M_1(3,5)\circ M(5,2)=CE\circ B=CEB$。又如$M_3(3,5)=0$，这是因$M_2(3,4)=CED$，而$M(4,5)=E$，故$M_2(3,4)\circ M(4,5)=CED\circ E=0$，其余类似可求。

（4）一般而言，$M_k(i,j)$表示路径的集合，例如$M_3(3,2)$是两条路径$CEAB$和$CDEB$的集合，表示从C到B存在两条长度为3的基本路径。它是由$M_2(3,1)\circ M(1,2)=CEA\circ B=CEAB$及$M_2(3,5)\circ M(5,2)=CDE\circ B=CDEB$得到的。

（5）$M_5=M_4*M$，其结果即为H回路的集合，但此时运用串乘时要注意，当两项的第一个结点相同时，其结果不能置0，而应串联起来。例如$M_5(1,1)=M_4(1,5)\circ M(5,1)=ABCDE\circ A=ABCDEA$。

§8.8 货郎担问题

货郎担问题也叫旅行商或巡回售货员问题，原始问题是这样提出的，一个推销员从驻地出发到预定的一些村镇去推销商品，那么他应选择怎样一条路线，使每一村镇都经过一遍最后回到原驻地而总的路程最短。

货郎担问题有着广泛的意义，它不仅是如何解决推销员的最优巡回路线问题，而且诸如公安执勤人员的最优巡回路线、流水作业生产线的顺序问题，装配线进度问题、数控机床的运行问题、以至机组人员的轮班安排、教师任课班级负荷分配等问题，都直接或间接与货郎担问题有关。因而引起人们的极大注意，也提出了多种不同的解法，但是到目前为止，还没有一个求解货郎担问题的有效算法，使这一问题成为组合优化中的著名难题。

货郎担问题可以用图论的语言来描述，构造一个图$G=(V,E)$，图中每一结点分别表示推销员的驻地和各个村镇，结点之间的边表示两地之间的通路，边上的权表示通路长度，于是货郎担问题就化归为在带权图中寻找一条通过每个结点至少一次的最短回路问题。

定义 8.9 在边带权图$G=(V,E)$中，经过每个结点一次且仅一次的权最小的回路，称为最优H回路。经过每个结点至少一次的权最小的回路称为最优货郎担回路。

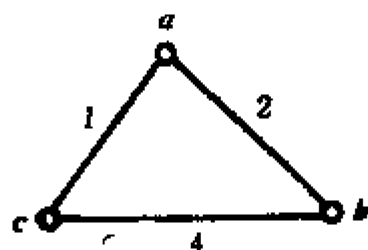


图 8.51

一般来说，最优货郎担回路与最优H回路是有区别的。如图8.51所示的带权图，最优H回路为(a,b,c,a)，权值为7，但最优货郎担回路有(a,b,a,c,a)，权值为6。由此可见、由于最优货郎担回路要求每个结点必须经过一次，但不一定只经过一次而是可以多于一次，因此在边带权不同的情况下，出现重复某些结点构成的回路权值反而小的现象，以致最优货郎担回路不同于最优H回路。为了区别这种情况，我们给出如下定义。

定义 8.10 在带权图$G=(V,E)$中，若G的任意一对结点u，v，对G的其余结点x，其权值$w(u,v)$均满足

$$w(u,v)\leqslant w(u,x)+w(x,v) \tag{8.6}$$

则称G为满足三角不等式的图。

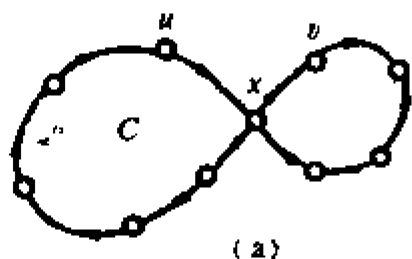
因此，图8.51所示的图不满足三角不等式。

定理 8.23 若带权图满足三角不等式，则它的最优货郎担回路与最优H回路相等(当存在H回路时)

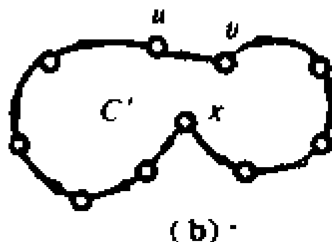
证：设最优货郎担回路为C，若C不是H回路，则至少有一个结点在C中出现不止一次，设这样的结点为x，现在我们沿着C的行走路线，把第一次到达x的先头结点记作u，x的后继结点记作v，如图8.52(a)所示，这时我们删除边(u,x)和(x,v)而增加边(u,v)，则回路C成为回路C'，如图8.52(b)所示。由于图满足三角不等式，C'的权不会增加，即$W(C')\leqslant W(C)$。但是C'含的边与C相比少了一条，而通过结点的次数少了一次，如果C'还含有重复结点，则继续上述过程直到无重复结点为止，最后得到经过各结点仅一次的最优货郎担回路，显然也是最优H回路。 ▌

在上述情况下，寻找最优货郎担回路的问题也就是寻找最优H回路的问题。

如果带权图$G=(V,E)$是H图但不满足三角不等式，前面的例子已说明，最优货郎担回路不是最优H回路，但是我们仍然可以将最优货郎担回路问题化归为最优H回路问题求解。方法是由给定的图$G=(V,E)$构造一个新的完全图$G'=(V',E')$，其中$V'=V$，E'中的每一条边(u,v)，它的权$w(u,v)$等于G中u与v之间的距离，由于G不满足三角不等式，u与v之间的距离可能不是边(u,v)上的权，而是u和v之间一条最短路径上的边权之和，为了便于查找，可以在G'中的相应边上标出这一路径。如图8.53所示的图G'就是按照上述方法将图8.51的图G改造而得。可以看到，G'中最优H回路和G中的最优货郎担回路是对应的，于是得到如下定理。



(a)



(b)

图 8.52

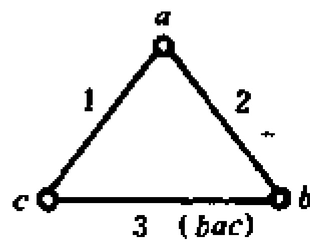


图 8.53

定理 8.24 G的最优货郎担回路的权和G'的最优H回路的权相同。

证：设C是G的一条最优货郎担回路，若C只经过每个结点一次且仅一次，显然它也是G'的最优H回路。否则设G'中与C在G中行走路线相同的回路为C'，那么C'的边不仅与C的边顺序对应，而且对应边的权也相等，这是因为如果对应边的权不相等，比如C'中的一条边(u,v)，它的权小于C中同一条边(u,v)的权，我们就可用C'中代表(u,v)的权的最短路径取代C中的边(u,v)，这就与C是G的最优货郎担回路相矛盾。

由于C'在G'中经过每个结点至少一次，现在考察经过次数多于一次的结点，设为x，令u为第一次到达x的先头结点，v为x的第一个后继结点，由于G'是满足三角不等式的完全图，因此用边(u,v)取代边(u,x)和(x,v)所得的回路仍然是最优货郎担回路，对所有重复结点都施行上述操作，最后得到的是G'的最优H回路，也是G的最优货郎担回路。 ▌

综上所述，不管哪一种情况，货郎担问题均可化归为最优H回路问题求解，这时的图

一般情况下应是带权完全图。

在一个边带权的完全图中寻找最优H回路，一种直觉的解法就是采用枚举法，即求出图的所有H回路，然后进行比较，权最小的H回路即为所求，这种方法表面看来似乎简单易行，但当图的结点数n较大时，其计算复杂性是难以接受的。由定理8.19可知，n阶完全图共有$(n-1)!/2$个不同的H回路，因此求出所有H回路的计算量是$O(n!)$级的，它随n的增加指数上升，当n较大时，即使采用大型高速计算机，也难以在较短时间内完成计算任务。例如设计算机的加法计算速率为10^8次/秒，n取不同值时，计算机需要的计算时间如表 8.2 所示。

表 8.2

n	$\sim n!$	计 算 时 间
12	4.8×10^8	5 秒
15	1.3×10^{12}	3 小时
20	2.4×10^{18}	800 年
50	3.0×10^{64}	10^{49} 年

由表可见，只有20个结点的完全图，计算时间竟需要 800 年，枚举算法效率之低可想而知。因此人们都在寻求计算量在多项式时间内的近似解法。

§8.9 货郎担问题的近似解法

当提出货郎担问题近似解的算法时，人们总是希望由算法求出的近似值，在保证计算量为多项式时间的前提下尽量接近精确值。设L为计算得到的近似值。L_0为精确值，当问题是求最小值时，令

$$1\leqslant L/L_0\leqslant\alpha$$

若问题是求最大值，则将L/L_0颠倒，于是α越接近 1，算法的精确度越高。

下面介绍近似解的几种算法，这里均假定图满足三角不等式条件，于是图的最优H回路也就是最优货郎担回路。

一、最邻近法

设图为完全无向图，算法步骤如下：

1．任意找一结点为始点，设为v_1，在其余$n-1$个结点中找一与v_1最邻近（即距离最小）的点作为初始通路。然后按步骤 2 逐点扩展通路。

2．设v_i是最新加到通路上的点，从不在通路上的所有点中找一与v_i最邻近的点v_j，把边(v_i,v_j)加到通路上。

3．重复步骤 2，直到图的所有结点都在路径上为止，设最后加入的点为v_n。

4．将边(v_n,v_1)加到路径上，即得所求的H回路。

例 8.13 对图8.54(a)所示的完全图，用最邻近算法，各步骤如图(b)～(e)所示，最后得到一H回路C，其权值为

$$W(C)=47$$

事实上，它并不是最优H回路，图(f)才是最优H回路，其权值为$W(H)=35$

用最邻近法计算产生的误差，主要原因在于最后两条边的加入是不容选择的，可以证明（见〔10〕）这一算法的α值为

$$\alpha=\frac{1}{2}\left(\lceil \ln n\rceil+1\right)$$

可见随着n的增加，α并无止境，这一算法虽然简单，但误差无法控制。

图 8.54

二、逐次修正法

算法步骤如下：

1. 在图中任意找一H回路C_0，设

$$C_0=v_1, v_2, \cdots, v_i, \cdots, v_j, \cdots, v_n, v_1$$

2. 对所有的i, j，$1<i+1<j<n$，若

$$w(v_i, v_j)+w(v_{i+1}, v_{j+1})<w(v_i, v_{i+1})+w(v_j, v_{j+1})$$

则在C_0中删去边(v_i, v_{i+1})和(v_j, v_{j+1})而加入边(v_i, v_j)和(v_{i+1}, v_{j+1})，形成新的回路C，即

$$C=v_1, v_2, \cdots, v_i, v_j, v_{j-1}, \cdots, v_{i+1}, v_{j+1}, \cdots, v_n, v_1$$

（见图8.55(a)和(b)）

3. 对C重复步骤2，直到条件不满足为止，最后得到的C即为所求。

例 8.14 对图8.56所示的完全图，用逐次修正法求最优H回路。

解： 任选一H回路C_0（见图8.57(a)）

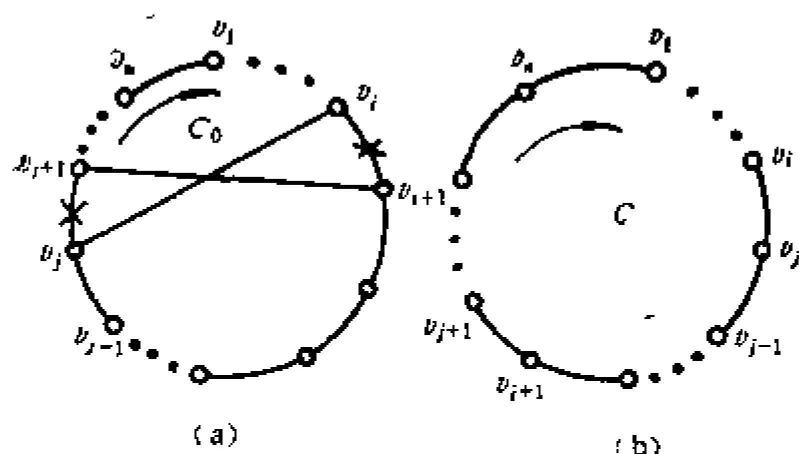


图 8.55

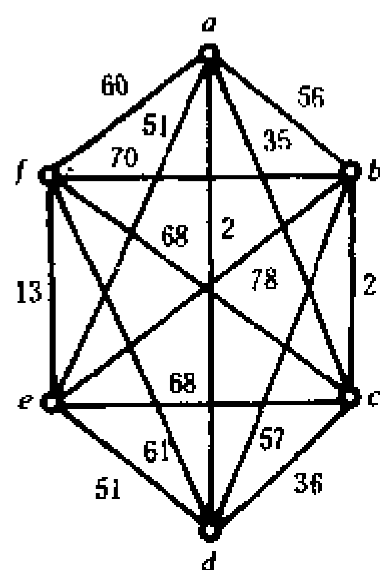


图 8.56

$$C_0=a,\ b,\ c,\ d,\ e,\ f,\ a$$

可以查到

$$w(a,\ d)+w(b,\ e)<w(a,\ b)+w(d,\ e)$$

故删除边 (a,b) 和 (d,e) 而加入边 (a,d) 和 (b,e)，得到新的回路C_1（见图(b)）

$$C_1=a,\ d,\ c,\ b,\ e,\ f,\ a$$

从 C_1 又可查到

$$w(a,\ e)+w(b,\ f)<w(f,\ a)+w(b,\ e)$$

故删除边 (f,a) 和 (b,e) 而加入边 (a,e) 和 (b,f)，得到新的回路 C_2（见图(c)）

$$C_2=a,\ d,\ c,\ b,\ f,\ e,\ a$$

从 C_2 又可查到

$$w(a,\ c)+w(e,\ d)<w(a,\ e)+w(c,\ d)$$

故删除边 (a,e) 和 (c,d) 而加入边 (a,c) 和 (e,d)，得到新的回路C_3（见图(d)）

$$C_3=a,\ c,\ b,\ f,\ e,\ d,\ a$$

在 C_3 上已找不到满足条件的边，故 C_3 即为所求的最优H回路，其权值为

$$W(C_3)=192$$

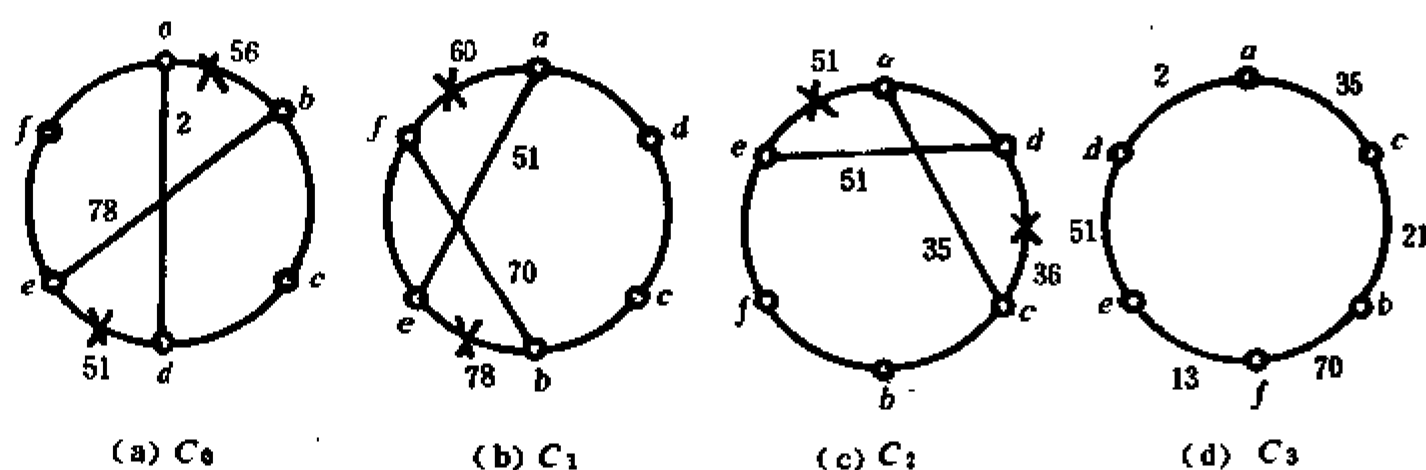


图 8.57

这一算法的误差界限，有时可用最小生成树加以说明。设C_0是图G的最优H回路，则对图的任一结点v，C_0-v是在$G-v$的H路径，因而也是$G-v$的生成树，由此可见，若T是$G-v$的一棵最小生成树，而且e_1和e_2是与v关联的边中权最小的两条边，则$W(T)+w(e_1)+w(e_2)$将是$W(C_0)$的一个下界，若用这一算法求出的H回路记作C，则可得到

$$W(T)+w(e_1)+w(e_2)\leqslant W(C_0)\leqslant W(C)$$

如图 8.56 所示的例子中，若取结点C作为v，可得 $G-C$ 的一棵最小生成树T如图8.58所示，与结点C关联的权最小的两条边为 (b,c) 和 (a,c)，此时

$$W(T)+w(b,\ c)+w(a,\ c)=122+21+35=178$$

因此最优H回路的权$W(C_0)$满足

$$178\leqslant W(C_0)\leqslant 192$$

三、重绕最小生成树法

所谓重绕生成树是用深度优先搜索法遍历生成树的边，但当退回时经过的边也计算在内，因此重绕生成树将对树的每条边经历两次因而得到一个回路。如图8.59所示的一棵生

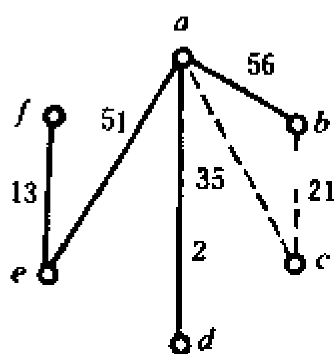


图 8.58

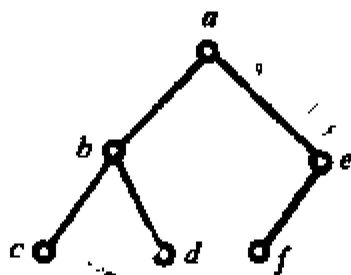


图 8.59

成树T，采用重绕生成树法得到的回路C_0为

$$C_0=(a, b, c, b, d, b, a, e, f, e, a)$$

本算法就是用重绕最小生成树得到回路C_0，再从C_0得到H回路，算法步骤如下：

1．求出完全图G的一棵最小生成树T

2．重绕最小生成树T得到回路C_0

3．按访问的先后次序检查C_0，将以后出现的重复结点删除（最后一个结点不能删除），得到一个结点不重复的回路C，即为所求的H回路。

例 8.15 用重绕最小生成树法求图8.60所示的完全图的最优H回路。

解：求出图G的一棵最小生成树T，如图中的粗线所示，以v_1为始点重绕生成树T得到回路C_0如下

$$C_0=(v_1, v_2, v_3, v_2, v_4, v_2, v_1, v_5, v_1, v_6, v_1)$$

依次序删除C_0中间重复出现的结点得到C

$$C=(v_1, v_2, v_3, v_4, v_5, v_6, v_1)$$

C即为所求的最优H回路。这里

$$W(C)=14,\ W(C_0)=16$$

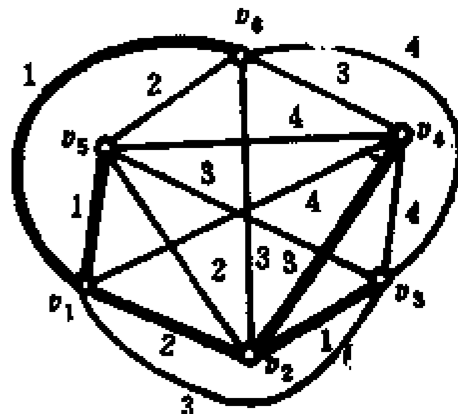


图 8.60

定理 8.25 对重绕最小生成树算法，有

$$\alpha<2$$

证：设T为G的最小生成树，$W(T)$表示它的权，H为G的一条最优H回路，$W(H)$表示它的权，H回路去掉任一条边即是G的生成树，因此必有

$$W(T)<W(H)$$

C_0是重绕T形成的回路，T的每一条边在C_0中重复出现一次，故C_0的权是T的权的二倍，即

$$W(C_0)=2W(T)<2W(H) \qquad (A)$$

算法中C是从C_0中删除重复结点而得到的H回路。删除重复结点实际上是用一条边取代两条边，例如在$C_0=\cdots v_j\cdots, v_i, v_j, v_k\cdots$，中$v_j$是重复结点，删除后成为$C=\cdots, v_j, \cdots, v_i, v_k\cdots$，即是用边$(v_i, v_k)$取代边$(v_i, v_j)$和$(v_j, v_k)$，因图满足三角不等式$w(v_i, v_k)\leqslant w(v_i, v_j)+w(v_j, v_k)$，故有

$$W(C)\leqslant W(C_0) \qquad (B)$$

由式（A）和（B）即得

$$W(C)/W(H)<2 \qquad \blacksquare$$

这一算法的计算复杂性主要在第1步和第2步，第一步是求图的最小生成树，如果我们采用$Prim$算法，其复杂性为$0(n^2)$级，第2步是用深度优先搜索法，其复杂性为0

$(max(n, |E|))$，可见这一算法的复杂性是低阶多项式级的，而 α 能保证在 2 以下，所以不失为一较好的算法。

如果对这一算法给予改进，还可以使 α 值降低，这就是下面介绍的算法。

四、最小权匹配算法

算法步骤如下：

1．求出图 G 的最小生成树 T

2．构造图 G'(G'是由 T 的所有奇次结点以及它们之间关联的边组成)，求出 G' 的最小权完全匹配 M。

3．将 M 的边加到 T 上得到欧拉图 G^*

4．求出 G^* 的欧拉回路 C_0

5．按访问的先后次序检查 C_0，将以后出现的重复结点删除(最后一个结点不能删除)，得到一个结点不重复的回路 C，即为所求的 H 回路。

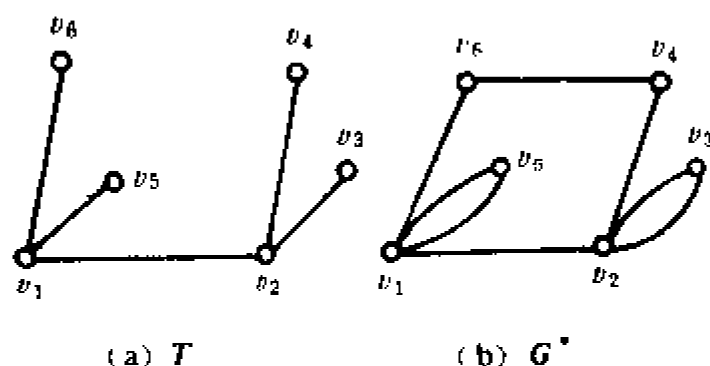
算法说明：

本算法与算法（三）比较，都是首先求出图的最小生成树，不同的是这里用第 2、3，4 步取代算法（三）的第 2 步。由于奇次结点的数目一定是偶数，所以 G' 含偶数个结点，因此一定能求得最小权完全匹配 M，将匹配边加到 T 上得到图 G^*，G^* 的所有结点都是偶次，因而是欧拉图，可以求出欧拉回路 C_0，以上各步计算的时间复杂性都是多项式级的。

例 8.16 如例 8.15，现将求出的最小生成树 T 重画出如图8.61(a)所示。这里 T 的6个结点都是奇次点，因而 $G'=G$，求出 G' 的最小权完全匹配

$$M=\{(v_1, v_5), (v_2, v_3), (v_4, v_6)\}$$

将 M 的边加到 T 上得到图 G^*，如图 8.61(b)所示。



(a) T (b) G^*

图 8.61

由 G^* 得到欧拉回路

$$C_0=(v_1, v_5, v_1, v_2, v_3, v_2, v_4, v_6, v_1)$$

依次删除 C_0 中间重复出现的结点得到 C

$$C=(v_1, v_5, v_2, v_3, v_4, v_6, v_1)$$

C 即为所求的最优 H 回路，这里

$$W(C)=12$$

定理 8.26 对最小权匹配算法，有

$$\alpha<\frac{3}{2}$$

证：在定理8.25的证明中已得出

$$W(C)\leqslant W(C_0)$$

这里 C_0 是由 T 和 M 的边组成的回路，故

$$W(C_0)=W(T)+W(M)$$

设 H 是图的最优 H 回路，因而

$$W(T) < W(H)$$

只要证明

$$W(M) \leqslant \frac{1}{2} W(H)$$

即有

$$W(C) \leqslant W(C_0) = W(T) + W(M) < \frac{3}{2} W(H)$$

定理即可得证。下面我们证明这一不等式。

已知H是G的最优H回路，它通过G的每一结点一次且仅一次，如图8.62中的实线所示，图中的实心圆点表示生成树T上的奇次结点，空心圆点则表示T上的偶次结点。

现在我们沿着H的轨迹构造一新的回路C'，即在H回路上如遇到空心圆点则绕过去（即用空心圆点前后两个实心圆点之间的边代替空心圆点的关联边）如图中的虚线所示，由于图满足三角不等式，故

$$W(C') \leqslant W(H)$$

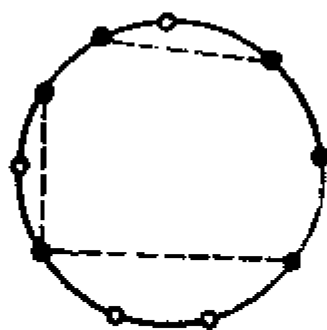


图 8.62

C'上结点全是T的奇次结点，数目为偶数，因而在C'上可以构造两个完全匹配M_1和M_2，两个匹配的边交替出现，且

$$W(C') = W(M_1) + W(M_2)$$

选其中一个权较小的匹配，不妨设为M_1，则

$$W(M_1) \leqslant \frac{1}{2} W(C') \leqslant \frac{1}{2} W(H)$$

在前面的算法中我们求出的匹配是T中奇次结点之间的最小权匹配，故M_1的权不会小于M的权，即

$$W(M) \leqslant W(M_1)$$

所以

$$W(M) \leqslant \frac{1}{2} W(H) \qquad \blacksquare$$

综观上述四种算法，在满足计算复杂性为多项式量级的条件下，最小权匹配法是较好的一种近似算法，目前还没有一个更为有效的近似算法，能使α值比定理8.26给出的α值更小。

§8.10 用分枝定界法解货郎担问题

上一节给出的算法都是货郎担问题的近似解，这一节我们讨论用分枝定界法求出精确的最优货郎担回路。在满足三角不等式的条件下，也就是求出最优H回路。

我们知道，H回路是通过图的每个结点一次且仅一次的回路，对于完全图来说，不同的H回路实际上是结点的不同排列，为了避免重复都指定一个结点作为排列的第一个结点，因此n阶完全无向图有$(n-1)!/2$个不同的排列，因而有$(n-1)!/2$个不同的H回路，对于n阶完全有向图，则有$(n-1)!$个不同的H回路。例如5个结点v_1, v_2, v_3, v_4, v_5的完全有向图，共有24个不同的H回路，即

$$v_1, v_2, v_3, v_4, v_5, v_1$$

$$v_1, v_2, v_3, v_5, v_4, v_1$$

$$\cdots$$

$$\cdots$$

$$v_1, v_5, v_4, v_3, v_2, v_1$$

我们可以构造一棵状态空间树来描述问题的解，如图 8.63 所示。这时问题的状态空间树是以结点 v_1 为根，高为 n 的一棵正则树，共有 $(n-1)!$ 片树叶，从树根到每一片树叶的路径，表示H回路的一个解。

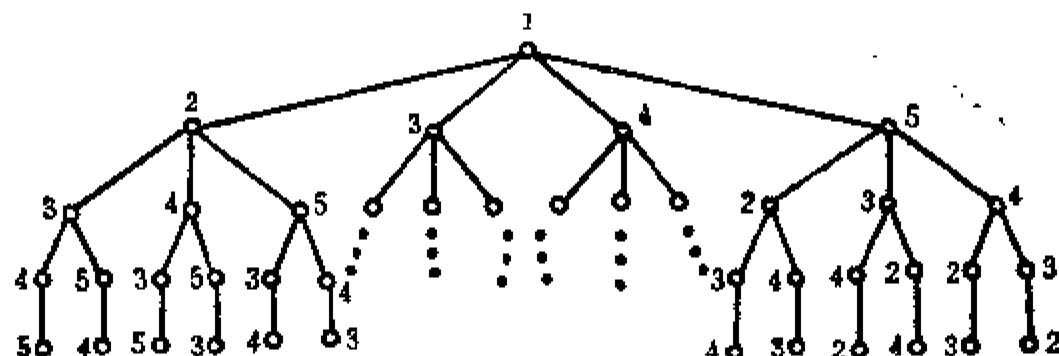


图 8.63

前面已讲过，当 n 较大时，企图遍历具有 $(n-1)!$ 片树叶的状态空间树，它的计算量是难以接受的。分枝定界法就是根据一些约束条件，只产生解的部分状态空间树，从而加速搜索过程，即通过计算一小部分可行解即可从中找到最优解，因面相对地减少了计算量。这里分枝和定界是算法的基础。现分述如下

一、定界

所谓定界，就是求出问题解的上、下界，通过当前得到的下界排出一些次优解，为最终获得最优解提示方向。

对于上界，可以任取一条H回路，令其权值为上界的初始值。不失一般性，下面我们以 n 阶完全有向图作为讨论对象。于是任选一条H回路：

$$C=(v_1, v_2, \cdots, v_n, v_1)$$

它的权

$$W(C)=\sum_{(v_i, v_j)\in C} w(v_i, v_j)$$

显然，最优H回路的权不会超过 $W(C)$，于是就令 $W(C)$作为可行解的上界。

对于下界的求法要复杂一些。它需要从图的距离矩阵着手给予定义。

在 §7.2 中已给出求图的距离矩阵的算法，对于带权完全图在满足三角不等式时，图的邻接矩阵也是图的距离矩阵。这里我们规定矩阵元素 $d_{ii}=\infty$，$v_i\in V$。例如下面是一个 5 阶带权完全有向图的距离矩阵D：

$$D=\begin{matrix} & v_1 & v_2 & v_3 & v_4 & v_5 \\ v_1 & \infty & 25 & 40 & 31 & 27 \\ v_2 & 5 & \infty & 17 & 30 & 25 \\ v_3 & 19 & 15 & \infty & 6 & 1 \\ v_4 & 9 & 50 & 24 & \infty & 6 \\ v_5 & 22 & 8 & 7 & 10 & \infty \end{matrix}$$

因为H回路是通过图的每个结点一次且仅一次的回路，对图的距离矩阵的每一行，任何一个H回路必然含该行的一个且仅一个元素，对列的情况也是一样。求图的最优H回路，相当于在距离矩阵中找出n个元素，使它们既不同行也不同列，而且它们之和为最小值。显然直接在矩阵上寻找是相当复杂的。为此我们将矩阵进行简化，并根据简化矩阵构造状态空间树。

将矩阵的行（或列）中各元素减去本行（或列）最小元素之值，称为对行（或列）的简化，从第i行（或第j列）各元素减去的最小值，称为第i行（或第j列））的约数，记作$r(i)$（或$r'(j)$）。若矩阵元素为非负值，矩阵的每一行和每一列都已简化，得到的矩阵称为原矩阵的简化矩阵，所以，简化矩阵的每一行每一列都至少有一个零元素。

以上面的距离矩阵D为例，经过各行简化后得到D_1，在D_1中，除第4列外每列都已有零元素，因此只须对第4列简化，最后得到D的简化矩阵D'如下。

$$D_1=\begin{array}{c|ccccc|c} & v_1 & v_2 & v_3 & v_4 & v_5 & r(i) \\ v_1 & \infty & 0 & 15 & 6 & 2 & 25 \\ v_2 & 0 & \infty & 12 & 25 & 20 & 5 \\ v_3 & 18 & 14 & \infty & 5 & 0 & 1 \\ v_4 & 3 & 44 & 18 & \infty & 0 & 6 \\ v_5 & 15 & 1 & 0 & 3 & \infty & 7 \end{array}
\qquad
D'=\begin{array}{c|ccccc|c} & v_1 & v_2 & v_3 & v_4 & v_5 & r(i) \\ v_1 & \infty & 0 & 15 & 3 & 2 & 25 \\ v_2 & 0 & \infty & 12 & 22 & 20 & 5 \\ v_3 & 18 & 14 & \infty & 2 & 0 & 1 \\ v_4 & 3 & 44 & 18 & \infty & 0 & 6 \\ v_5 & 15 & 1 & 0 & 0 & \infty & 7 \\ r'(j) & 0 & 0 & 0 & 3 & 0 & \end{array}$$

矩阵简化后，称各行、列约数之和

$$r=\sum_{i=1}^{n} r(i)+\sum_{j=1}^{n} r'(j) \tag{8.7}$$

为原矩阵D的约数。如上例

$$r=(25+5+1+6+7)+3=47$$

若C是简化前的一个H回路，矩阵简化后，这个回路的权必减小而有如下定理

定理 8.27 设$D=(d_{ij})$是给定图的距离矩阵，$D'=(d'_{ij})$是D的简化矩阵，若C是G的任何一条H回路，则有

$$\sum_{\langle v_i,v_j\rangle\in C} d_{ij}=\sum_{\langle v_i,v_j\rangle\in C} d'_{ij}+r \tag{8.8}$$

证：因任一H回路有n条边，每一条边$\langle v_i,v_j\rangle$仅与D中的一个元素d_{ij}对应，矩阵简化后，$\langle v_i,v_j\rangle$仅与D'中的一个元素d'_{ij}对应，简化时第i行减去约数$r(i)$，第j列减去约数$r'(j)$，故

$$d'_{ij}=d_{ij}-r(i)-r'(j)$$

即

$$d_{ij}=d'_{ij}+r(i)+r'(j)$$

对所有的$\langle v_i,v_j\rangle\in C$均成立。由于不存在同一行（列）中有两个元素对应的边都在回路C中，故

$$\sum_{\langle v_i,v_j\rangle\in C} d_{ij}=\sum_{\langle v_i,v_j\rangle\in C}(d'_{ij}+r(i)+r'(j))$$

$$=\sum_{\langle v_i,v_j\rangle\in C} d'_{ij}+\sum_{i,j=1}^{n}(r(i)+r'(j))$$

$$= \sum_{\langle v_i, v_j \rangle \in C} d'_{ij} + r$$ ▮

这一定理告诉我们，若C是原有矩阵D中形成的最优H回路，则在简化矩阵D'中仍然是最优H回路。其次，若D'中各行各列的零元素构成一条H回路，则这条回路就是最优H回路，这是因为零元素在所在的行或列中是对应最小权的边，由此可知，任何一条H回路的权不可能小于r，因而r就是矩阵简化前H回路权的下界。

若矩阵简化后，零元素还不能形成H回路，我们可以从这些元素中选择一些边构成回路，为此我们可以构造一棵二元树，这就是下面要介绍的分枝。

二、分枝

二元树是这样构造的，结点表示H回路的集合，根结点表示所有H回路的集合，标(i,j)的结点表示含$\langle v_i,v_j\rangle$边的所有H回路构成的子集，标$(\overline{i,\ j})$的结点则表示不含$\langle v_i,v_j\rangle$边的所有H回路构成的子集，若结点(i,j)继续分枝，下面标(k,l)的点表示既含$\langle v_i,v_j\rangle$边又含$\langle v_k,v_l\rangle$边的所有H回路构成的子集，而标$(\overline{k,\ l})$的点则表示含$\langle v_i,v_j\rangle$边但不含$\langle v_k,v_l\rangle$的所有H回路构成的子集，如图8.64所示。如果继续分枝下去，则子集将越分越小。这时须用下界进行检查，即在分枝过程中，当某一回路子集的下界大于另一子集的下界时，则前者对应的结点不再分枝。

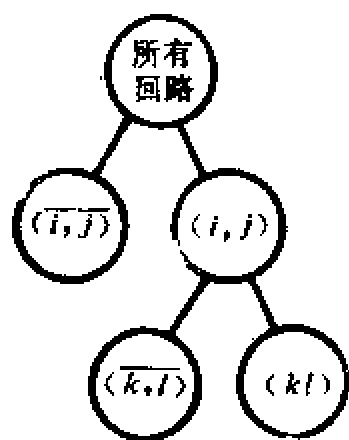


图 8.64

一般情况下，先选零元素$d'_{ij}=0$所对应的边$\langle v_i,v_j\rangle$进入回路，选定之后，可以删除所在行列的元素（或将这些元素都置∞），在以后的讨论中不再考虑，此外，当$\langle v_i,v_j\rangle$选入后，为了避免以后可能将边$\langle v_j,v_i\rangle$也选入回路而形成$(\langle v_i,v_j\rangle,\langle v_j,v_i\rangle)$于回路，这时应置$d'_{ji}=\infty$，这种处置应随每一条边的选入一同进行。

在简化矩阵D'中，每一行列都有零元素，应选哪一个零元素所对应的边进入回路，通常有如下两种选择

（1）在所有$d'_{ij}=0$的元素中，选择使得

$$\Delta(i,j)=\min_{k\neq j}\{d'_{ik}\}+\min_{k\neq i}\{d'_{kj}\}$$

值最大的所对应的边$\langle v_i,v_j\rangle$。

这是因为当边$\langle v_i,v_j\rangle$选入回路后，这条边所在的行和列的其他边都不能再进入回路，这样选择时，与它同一行的最小边和同一列的最小边的权和是所有$d'_{ij}=0$的边中最大的，这就保证在去掉第i行第j列的元素后，权和的下界上升较慢。

（2）在所有$d'_{ij}=0$的元素中，选择使得

$$S(i,j)=\sum_{\substack{k\neq j\\ d'_{ik}\neq\infty}} d'_{ik}+\sum_{\substack{k\neq j\\ d'_{kj}\neq\infty}} d'_{kj}$$

为最大的所对应的边$\langle v_i,v_j\rangle$。

这种选法其目的与（1）相同，

现在以前面给出的矩阵D及简化矩阵D'为例，说明分枝过程如下。

在矩阵D'中有 6 个零元素：$d'_{12},d'_{21},d'_{35},d'_{45},d'_{53},d'_{54}$，我们采用选择方法(1)，于是有

$$\Delta(1,2)=2+1=3,\quad \Delta(2.1)=12+3=15$$
$$\Delta(3,5)=2+0=2,\quad \Delta(4,5)=3+0=3$$
$$\Delta(5,3)=0+12=12,\quad \Delta(5,4)=0+2=2$$

其中 $\Delta(2,1)=15$ 最大，故选择边$\langle v_2,v_1\rangle$，于是从根结点（表示所有回路集合）引出两个分枝，一个引向结点（2，1），表示含边$\langle v_2,v_1\rangle$的所有回路构成的子集，一个引向结点$(\overline{2,\ 1})$，表示不含边$\langle v_2,v_1\rangle$的所有回路构成的子集。

对结点（2,1），我们将矩阵 D' 中的第 i 行和第 j 列的元素全都置∞（或将这两行删除）并置 $d_{12}=\infty$，得到对应于该点的矩阵及简化矩阵如下：

	v_1	v_2	v_3	v_4	v_5
v_1	∞	∞	15	3	2
v_2	∞	∞	∞	∞	∞
v_3	∞	14	∞	2	0
v_4	∞	44	18	∞	0
v_5	∞	1	0	0	∞

简化 $\Longrightarrow$

	v_1	v_2	v_3	v_4	v_5	$r(i)$
v_1	∞	∞	13	1	0	2
v_2	∞	∞	∞	∞	∞	
v_3	∞	13	∞	2	0	0
v_4	∞	43	18	∞	0	0
v_5	∞	0	0	0	∞	0
$r'(j)$		1	0	0	0	3

在简化过程中得到约数为 3，加上第一次简化的约数为 47，于是对应于该点的约数（下界）为 $r=47+3=50$

对结点$(\overline{2,\ 1})$，我们将 D' 中的 d'_{21} 置∞然后进行简化得到简化矩阵如下：

	v_1	v_2	v_3	v_4	v_5
v_1	∞	0	15	3	2
v_2	∞	∞	12	22	20
v_3	18	14	∞	2	0
v_4	3	44	18	∞	0
v_5	15	1	0	0	∞

	v_1	v_2	v_3	v_4	v_5	$r(i)$
v_1	∞	0	15	3	2	0
v_2	∞	∞	0	10	8	12
v_3	15	14	∞	2	0	0
v_4	0	44	18	∞	0	0
v_5	12	1	0	0	∞	0
$r'(j)$	3	0	0	0	0	15

由此得 $r=47+15=62$，如图 8.65(a)所示。由子树的左儿子的下界大于右儿子的下界，因此暂停对左儿子的分枝而对右儿子继续分枝。

对应结点（2,1）的简化矩阵中，有 6 个零元素 d_{15}，d_{35}，d_{45}，d_{52}，d_{53}，d_{54}，重复前面的选择方法得出 d_{45} 所对应的边$\langle v_4,v_5\rangle$，于是由点（2,1)引出两个分枝，一到点(4,5)，另一到点$(\overline{4,\ 5})$，如此继续下去即得到图 8.65(b)的状态树。

可以看到此时右分枝的两片树叶的下界已大于左分枝，继续分枝下去下界还要增加，因此应回溯对点$(\overline{2,\ 1})$进行分枝，得到图 8.66 所示的状态树。

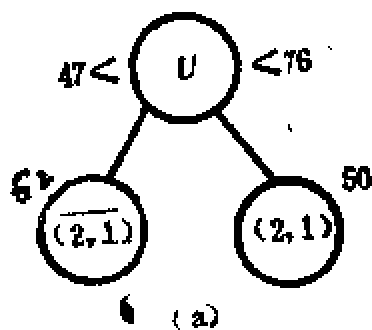
由图可见，结点（1,2）是一个解结点，由树根到此结点经过的边是

$$4\to1,\ 2\to3,\ 3\to5,\ 5\to4,\ 1\to2$$

因而得到最优 H 回路为

$$v_1,v_2,v_3,v_5,v_4,v_1$$
$$W(C)=62$$

货郎担问题的分枝定界算法



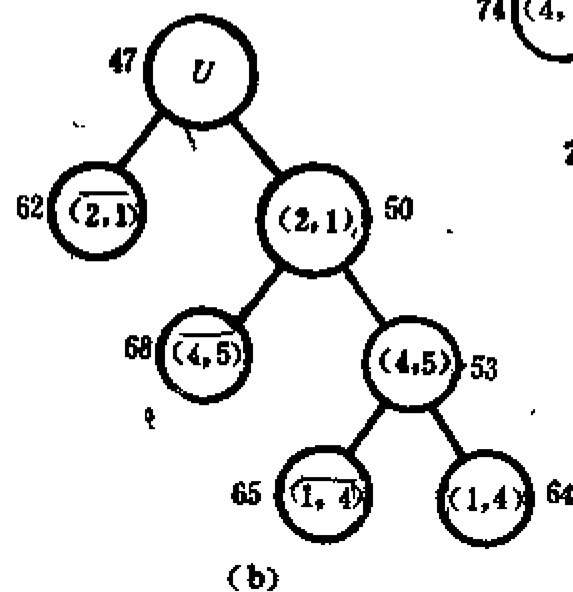


图 8.65

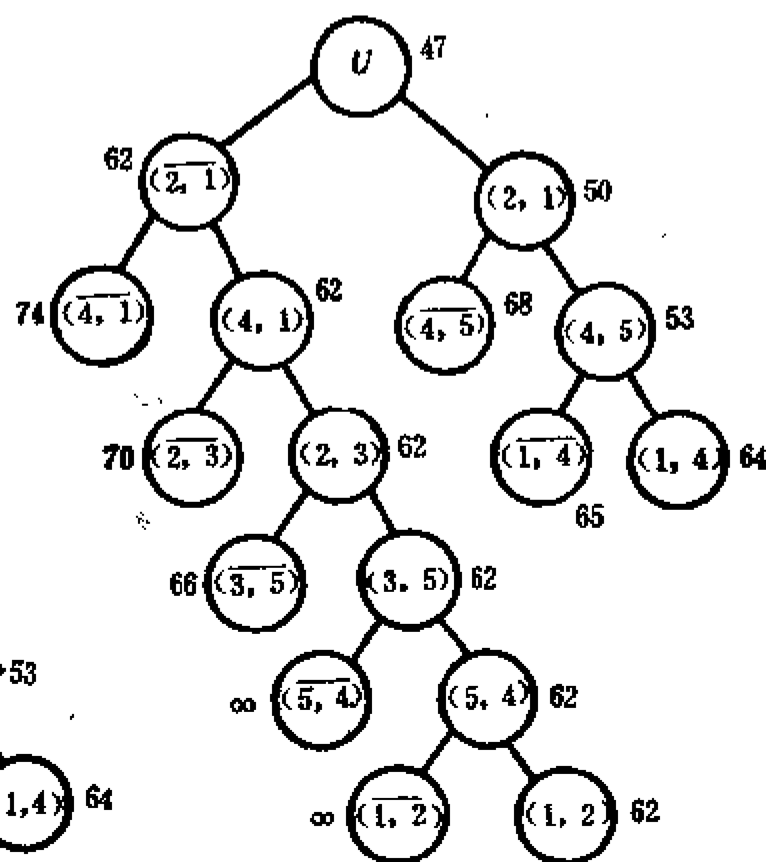


图 8.66

1. $E \leftarrow 1$, $L \leftarrow \phi$, PARENT(1)←0, $U \leftarrow \infty$
2. 设A_1是距离矩阵D的简化矩阵，r_1是D的约数
3. $\tilde{C}(1) \leftarrow r_1$, $N \leftarrow 0$
4. While $\tilde{C}(E) < U$ *do*

   begin
5. 设$\langle i,j \rangle$是E的简化矩阵A_E中满足A_E $(i,j)=0$且使得$\Delta_E(i,j)$最大的一条边
6. $ET(N+1) \leftarrow \langle i,j \rangle$; $ET(N+2) \leftarrow \langle 0,0 \rangle$
7. $\bar{A}_{N+1} \leftarrow A_E$; $\bar{A}_{N+2} \leftarrow A_E$
8. 将$\bar{A}_{N+1}$的第i行及第j列的一切元素置∞，并且$\bar{A}_{N+1}(j,i) \leftarrow \infty$
9. 将$\bar{A}_{N+1}$简化成$\bar{A}_{N+1}$，设约数为r_{N+1}
10. $\tilde{C}(N+1) \leftarrow \tilde{C}(E)+r_{N+1}$; PARENT$(N+1) \leftarrow E$
11. if $N+1$是一个解结点且$\tilde{C}(N+1) < U$
12. then $U \leftarrow \tilde{C}(N+1)$; ans$\leftarrow N+1$
13. else if $N+1$不是解结点且$\tilde{C}(N+1) < U$
14. then 将 $(N+1)$ 加入表L中
15. $\bar{A}_{N+2}(i,j) \leftarrow \infty$
16. 将$\bar{A}_{N+2}$简化成A_{N+2}，设约数为r_{N+2}
17. $\tilde{C}(N+2) \leftarrow \tilde{C}(E)+r_{N+2}$; PARENT$(N+2) \leftarrow E$
18. if $\tilde{C}(N+2) < U$ then 将 $(N+2)$ 加入L中
19. $N \leftarrow N+2$
20. 从L中删去$\tilde{C}(X) \geqslant U$的一切活结点X
21. *if* $L \neq \phi$ then 从L中选出有最小下界函数值$\tilde{C}(X)$的结点X赋给E，并从L中删除这个结点。
22. else$\tilde{C}(E) \leftarrow \infty$

```
    end
23. Print ('Least cost=',U)
24. While ans≠0 do
    begin
25. Print(ET(ans))
26. ans←PARENT(ans))
    end
```

算法说明：

U为上界值，其初值置∞，$\tilde{C}(X)$为结点X的下界函数，L为活结点表，不可能产生解的结点称为死结点，只要有一个儿子不是死结点，就将此结点加入活结点表L中。E为当前扩展结点，$PAREN(X)$为X的父亲，ans为回答结点。

在算法中，先行后列和先行后行的简化顺序产生的简化矩阵，一般来说是不同的，如果最优H回路是唯一的，答案不会改变，否则不同的简化顺序可能会得出不同的最优H回路。

分枝定界法解货郎担问题的计算复杂性，在最坏情况下是$O(n^2 2^n)$，因为算法不可避免地要在整个状态空间树上进行搜索，即使产生部分状态树，生成的结点个数仍然是$O(2^n)$级的，对每个生成结点计算一个简化矩阵，需要时间$O(n^2)$，但是，如果定界函数选得好，可以得到一个小得多的常数因子。

习题与思考题

1．判定图 8.67(a)，(b)的图能否一笔画出。

2．求出图 8.68 所示的图的欧拉回路（按寻迹路线给边标号）

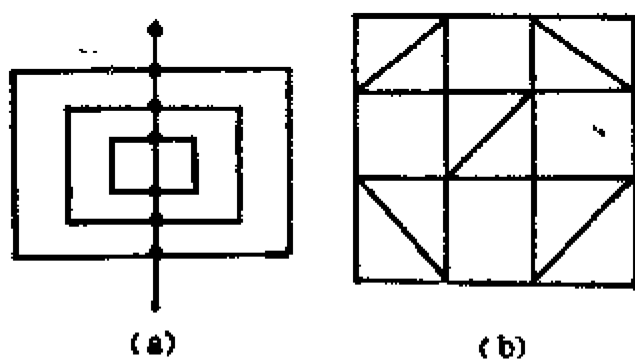


图 8.67

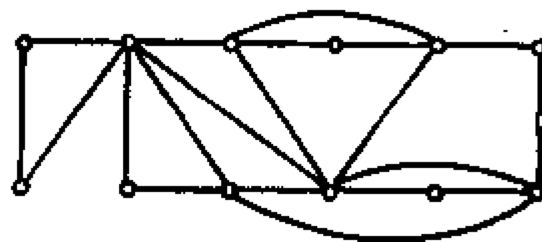


图 8.68

3．构造一个欧拉图，它的结点数n和边数m满足如下条件

（a）n，m的奇偶数相同

（b）n，m的奇偶数相反

如果不可能，说明原因。

4．确定n取怎样的值，完全图K_n是E图。

5．设G是具有k个奇次点的图，问至少加多少条边到G中，才能使得到的图是E图。

6．图 8.69 表示一座房子的平面图，问是否存在一条通过各个门一次且仅一次的通路。

7．根据给出的算法，用一高级语言编出求欧拉回路的程序上机调试通过。

8．试用 Fleury 算法求图 8.70 所示图的欧拉回路。

9．试用奇偶点作业法求图 8.71 所示的网络的最优投递回路。

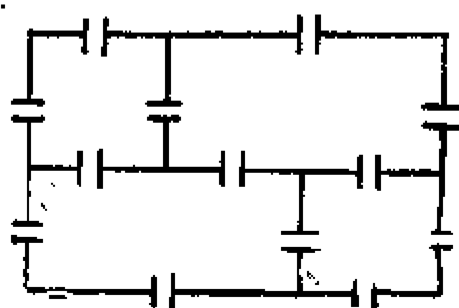


图 8.69

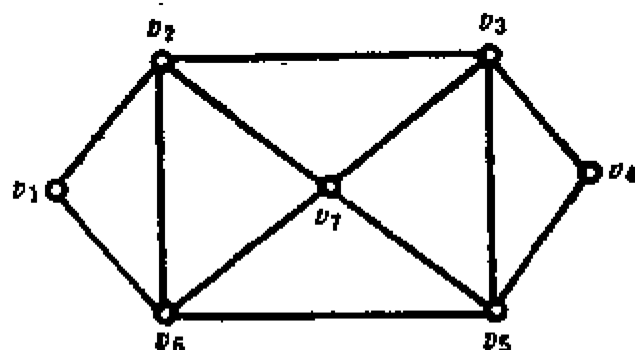


图 8.70

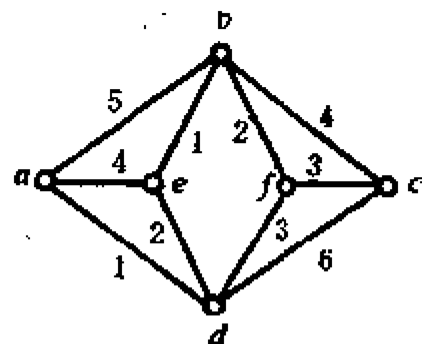


图 8.71

10．试用 Edmonds 算法解第 9 题。

11．证明：每一个有向欧拉图（无孤点）是强连通的。构造一个反例说明其逆命题不成立。

12．证明：若 E 是一个有向欧拉图，E_1 是 E 的真子图而且也是一个有向欧拉图，那么

$$E \oplus E_1 = E - E_1$$

是一个有向欧拉图。

13．试用逆向寻迹法求图 8.72 所示的有向图的有向欧拉回路。

14．求输出四个二进制码信息的鼓轮设计。

15．找出一种 9 个 a，9 个 b，9 个 c 的圆形排列，使由字母 $\{a, b, c\}$ 组成的，长度为 3 的 27 个字母仅出现一次。

16．求一个由 7 个 0 和 7 个 1 组成的循环序列，使得除 0000 和 1111 外的所有 4 位二进制数都作为这个序列的一段而出现。

17．对于图 8.73(a)，(b)，(c)，(d)所示的 4 个图，指出哪一个是欧拉图，哪一个是哈密尔顿图。

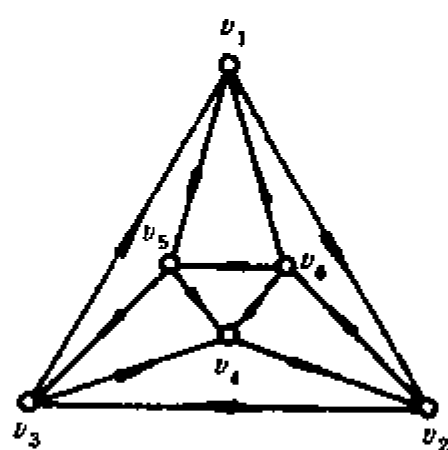


图 8.72

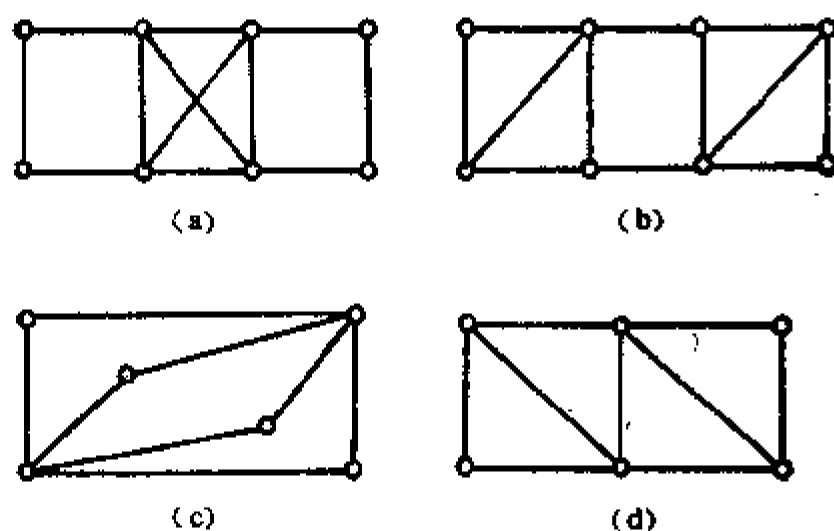


图 8.73

18．画一个图，使它

(a) 有一条欧拉路径但没有哈密尔顿路径。

(b) 有哈密尔顿路径但没有欧拉路径。

(c) 既没有欧拉路径也没有哈密尔顿路径。

19．证明：若一个图 G 中存在一个结点 v 有 $deg(v)=1$，那么 G 一定不是 H 图。

20. 设C是连通图G的一条回路，若删去C的任一条边得到的都是G的基本路径，证明G是H图

21. n个结点的简单图，若它的边数至少是$\frac{1}{2}(n-1)(n-2)+2$，证明它必有一条H回路。

22. 图8.74所示的两图中有没有哈密尔顿路径。

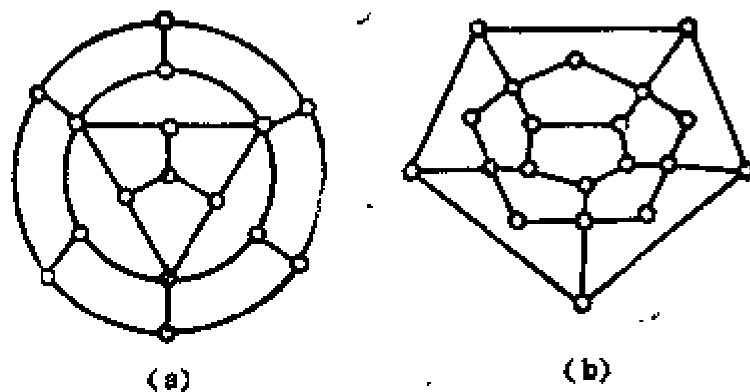


图 8.74

23. 设G是一个完全偶图$K_{m,n}$,试证:

(a) 若$m=n$，则G是H图

(b) 若$m\neq n$，则G为非H图

24. 证明：若G是自补图，则G有H路径。

25. 教室里有5排椅子，每排5张，每张椅子坐一个学生，如果一周后每个学生都必须与他相邻的某一个同学（前后左右）交换位置，问应当怎样换？

26. 求出图8.75中有向图的全部有向H回路。

27. 试用标记法判定图8.76所示的图是否存在H路径。

28. 试用回溯法求图8.77中无向图的所有H回路。

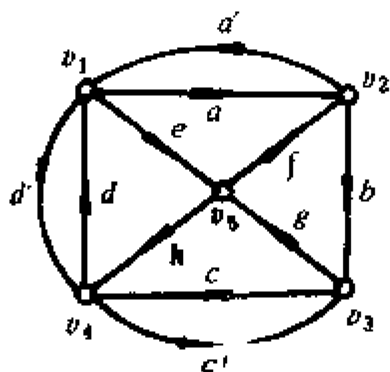
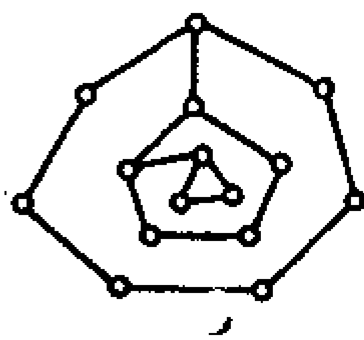


图 8.75


图 8.76

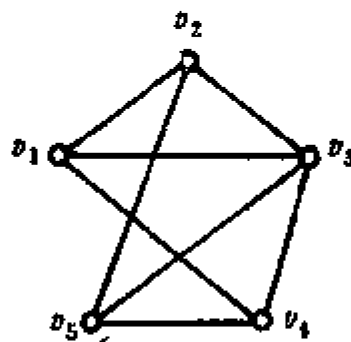


图 8.77

29. 试用矩阵法计算第28题

30. 分别用最邻近法、逐次修正法、重绕最小生成树法和最小权匹配算法求图8.78带权完全无向图的最优H回路。

31. 对图8.79所示的带权有向图，求最优H回路。

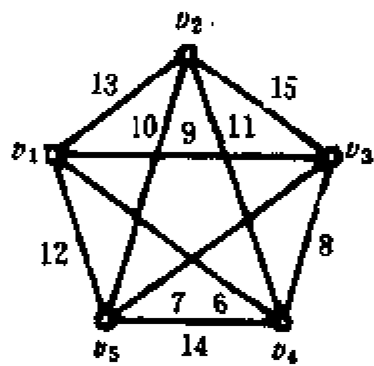


图 8.78

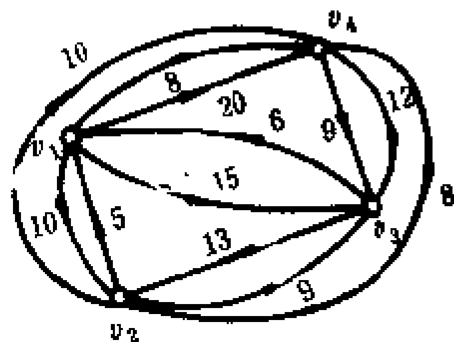


图 8.79

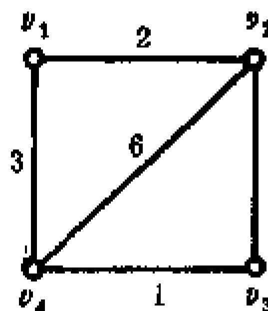


图 8.80

32. 对图8.80所示不满足三角不等式的带权网络，试求最优货郎担回路。

33. 设G是满足三角不等式的带权完全图，T是G的一棵最小生成树，C是最优H回

路，证明：$W(C) \leqslant 2(W)T$

34．用分枝定界法解下面的货郎担问题，这里D是5个结点v_1，v_2，v_3，v_4，v_5的距离矩阵

$$
D = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array}
\begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\
\begin{bmatrix} \infty & 34 & 2 & 50 & 59 \\ 34 & \infty & 36 & 68 & 67 \\ 2 & 36 & \infty & 51 & 60 \\ 50 & 68 & 51 & \infty & 13 \\ 59 & 67 & 60 & 13 & \infty \end{bmatrix} \end{array}
$$

35．试改变简化矩阵的简化顺序，重作第34题，并比较所得结果。

36．送入机械加工车间的每个工件都必须经过四种机械的加工，加工的先后次序可以任意，但在某一工序机台上的调装时间决定于前一加工工序。各工序间的调装时间如右表所示，试确定最优的加工顺序。

到 / 从	机台 A	B	C	D
机台A	＼	15	20	5
B	30	＼	30	15
C	25	25	＼	15
D	20	35	10	＼

第九章　网络的流

§9.1　流与切割

在第二章我们曾把带权图统称为网络，图可分为有向图与无向图两大类，带权图又可分边带权、点带权或边、点都带权，因此网络的涵意是广泛的，在对各种网络的分析中，网络的流是研究网络的一重要课题，寻求网络流的最大值，更是课题的主要任务，它在交通运输、信息传递等方面，有着广泛的应用。本章以运输网络为背景，以单发点、单收点的有向边权网络为重点，在理论上进行较为详尽的分析，给出求最大流的算法，在此基础上，进一步讨论多发点、多收点网络及无向网络，并将求最大流算法应用到求偶图的最大匹配问题上，最后讨论最小费用流。

一、单发点单收点有向网络

定义 9.1　设$N=(V,E)$是一连通且无自环的有向图，如满足以下条件

（1）有且仅有一个结点 s（或记作 v_s），它的引入次数为零。

（2）有且仅有一个结点 t（或记作 v_t），它的引出次数为零。

（3）任一条边 $\langle v_i, v_j\rangle \in E$ 均有一非负数的权。

则称N为一个网络。

常称引入次数为零的结点 s 为网络N的源或发点，称引出次数为零的结点 t 为网络N的汇或收点，而将边带的权称为该边的容量。边 $\langle v_i, v_j\rangle$ 的容量，记作 $C(i,j)$，一般说来 $C(i, j)\neq C(j, i)$

我们这里所定义的网络，是各种交通运输、信息传递系统的数学模型。边的容量在不同的问题上具有不同的物理意义，例如若N是一个交通运输网络，则 s 表示发送站，t 表示接收站，其余结点表示中间转运站，边的容量表示这条通路上能承担的最大运输量。

二、网上的流

网络上的流从发点 s 发出，通过网上的各条途径流入收点，流过边 $\langle v_i, v_j\rangle$ 的流量，记作 $f(i, j)$，称为边 $\langle v_i, v_j\rangle$ 的流。显然，边 $\langle v_i, v_j\rangle$ 上的流量不应超过边的容量，即

$$0\leqslant f(i, j)\leqslant C(i, j) \qquad (9.1)$$

例如网络是一个油路运输系统，边的容量表示这段管道允许流过的最大流量，而流则表示这段管道当前实际的流量。

定义 9.2　满足式（9.1）的网络称为相容网络。

如图 9.1 所示的网络N，每条边上标的第一个数表示该边的容量，第二个数表示当前流过该边的流量。即

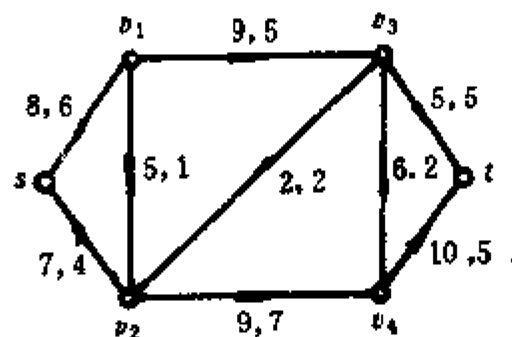


图 9.1

$$C(s,1)=8,\ f(s,1)=6,\ \cdots$$

定义 9.3 网络N上的某条边$\langle v_i, v_j\rangle$，若它的流量等于容量，即

$$f(i,j)=C(i,j)$$

则称该边为饱和边，否则称为非饱和边。

如图 9.1，边$\langle v_3, t\rangle$，$\langle v_3, v_2\rangle$当前为饱和边，其余均为非饱和边

定义 9.4 网络N的每一条边上流的集合称为网上的流，记作F，即

$$F=\{f(i,j)\mid \langle v_i, v_j\rangle \in E\}$$

例如图 9.1，网络N上的流为

$$F=\{6,4,1,5,2,7,5,2,5\}$$

对于任意结点v_i，以$\sum\limits_j f(i,j)$表示从v_i输出流的总和，而以$\sum\limits_k f(k,i)$表示输入v_i流的总和。

定义 9.5 对网络的任一中间结点v_i（即$v_i \neq s, v_i \neq t$）如有

$$\sum_j f(i,j)=\sum_k f(k,i) \tag{9.2}$$

则称网络N是守恒的。

守恒网络是一种无损耗网络，是实际运输网络的理想化，这时发点输出流的总和等于输入收点流的总和，其值记作$f(s,t)$，即

$$f(s,t)=\sum_j f(s,j)=\sum_k f(k,t)$$

定义 9.6 若网络N满足式（9.1）和（9.2）两个条件，则称F是网络N上的可行流，并称$f(s,t)$为可行流F的值。

如图9.1，此时网上的流就是可行流，它的值$f(s,t)=10$。

显然，每一个网络N至少有一个可行流，因为对任一条边$\langle v_i, v_j\rangle \in E$，如果令$f(i,j)=0$，由此得的流必然满足式（9.1）和(9.2)，因而是可行流。我们称这样设置的可行流为零流。

研究网络流的任务，在于寻求它的最大流，即在满足式（9.1）和（9.2）的条件下，求$f(s,t)$的最大值，因（9.1）式是个不等式，这个问题是一个典型的线性规划问题，但用图论的方法来解决，更为简洁有效。

三、切割

定义 9.7 对网络$N=(V,E)$，设V_1和V_2是V的两个不相交的非空子集，用(V_1, V_2)表示始于V_1中的结点终止于V_2中的结点的边的集合，$f(V_1,V_2)$表示这些边上流的总和，即

$$(V_1,V_2)=\{\langle v_i, v_j\rangle \mid v_i\in V_1, v_j\in V_2\}$$

$$f(V_1,V_2)=\sum_{\langle v_i, v_j\rangle\in(V_1,V_2)} f(i,j)$$

如图 9.1 所示，设$V_1=\{v_1, v_2\}$，$V_2=\{v_3, v_4\}$，则$(V_1, V_2)=\{\langle v_1, v_3\rangle, \langle v_2, v_4\rangle\}$，$f(V_1,V_2)=5+7=12$。

定义 9.8 网络$N=(V,E)$的一个分离发点s和收点t的切割K，是边的集合$(V_1,$

$\bar{V}_1$)，这里 $V_1 \subset V$, $\bar{V}_1 = V - V_1$，且 $s \in V_1$，$t \in \bar{V}_1$

例 9.1 如图 9.1 的网络 N

（1）若取 $V_1 = \{s, v_1, v_2\}$，则 $V_1 = \{v_3, v_4, t\}$ 此时 $K_1 = (V_1, \bar{V}_1) = \{\langle v_1, v_3 \rangle, \langle v_2, v_4 \rangle\}$ 是 N 的一个切割。

（2）若取 $V' = \{s, v_1\}$，则 $\bar{V}' = \{v_2, v_3, v_4, t\}$，此时 $K_2 = (V', \bar{V}') = \{\langle s, v_2 \rangle, \langle v_1, v_2 \rangle, \langle v_1, v_3 \rangle\}$ 也是 N 的一个切割。

因此，取不同的 V_1（必须 $s \in V_1$, $t \in V - V_1$）将得到不同的切割，但它们都是分离发点 s 和收点 t 的。

必须把这里讲的切割与第三章讲的割集区别开来，切割是针对流向而言，即切断了所有从 V_1 流向 $\bar{V}_1$ 的流，但图不一定被分离成两个分支，而割集却是从分割图来定义的。

定义 9.9 切割 K 的容量，是它的各边容量之和，记作 $C(V_1, \bar{V}_1)$ 或 $C(K)$，即

$$C(K) = \sum_{\langle v_i, v_j \rangle \in (V_1, \bar{V}_1)} C(i, j) \tag{9.3}$$

如例 9.1，因 $C(1,3) = 9$，$C(2,4) = 9$，$C(s, 2) = 7$，$C(1, 2) = 5$，则

$$C(K_1) = 9 + 9 = 18$$

$$C(K_2) = 7 + 5 + 9 = 21$$

定理 9.1 对网络 $N = (V, E)$ 的任一切割 $(V_1, \bar{V}_1)$，恒有

$$\begin{aligned} f(s, t) &= f(V_1, \bar{V}_1) - f(\bar{V}_1, V_1) \\ &= \sum_{\substack{i \in V_1 \\ j \in \bar{V}_1}} f(i, j) - \sum_{\substack{j \in \bar{V}_1 \\ i \in V_1}} f(j, i) \end{aligned} \tag{9.4}$$

证：根据流的条件，有

$$f(s, t) = \sum_v f(s, v) - \sum_v f(v, s)$$

其中 $\sum_v f(s, v)$ 表示从发点 s 流出的流之和，$\sum_v f(v, s)$ 表示流入发点之流的和。

此外对任一结点 $u \in V_1$（$u \neq s$）均有

$$\sum_v f(u, v) - \sum_v f(v, u) = 0$$

根据上面两个方程，对 V_1 中的所有结点（包括 s）的流求和，得到

$$f(s, t) = \sum_{u \in V_1} \left(\sum_v f(u, v) - \sum_v f(v, u) \right)$$

因为

$$\sum_{u \in V_1} \sum_v f(u, v) = \sum_{\substack{u \in V_1 \\ v \in V_1}} f(u, v) + \sum_{\substack{u \in V_1 \\ v \in \bar{V}_1}} f(u, v)$$

而

$$\sum_{u \in V_1} \sum_v f(v, u) = \sum_{\substack{u \in V_1 \\ v \in V_1}} f(v, u) + \sum_{\substack{u \in V_1 \\ v \in \bar{V}_1}} f(v, u)$$

显然有

$$\sum_{\substack{u\in V_1\\ v\in V_1}} f(u,v)=\sum_{\substack{u\in V_1\\ v\in V_1}} f(v,u)$$

故

$$f(s,t)=\sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} f(u,v)-\sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} f(v,u)$$

$$=f(V_1,\bar{V}_1)-f(\bar{V}_1,V_1) \quad ∎$$

这一定理告诉我们：从网络N的发点s到收点t的流值，等于任意一个切割中流的净值，即从切割的V_1到$\bar{V}_1$的流减去从$\bar{V}_1$到V_1的流的总值。

推论 1：任一网络的流值均不可能超过它的任一切割的容量，即

$$f(s,t)\leqslant c(V_1,\bar{V}_1) \tag{9.5}$$

证：由定理知，对任一切割$(V_1,\bar{V}_1)$均有

$$f(s,t)=\sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} f(u,v)-\sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} f(v,u)$$

$$\leqslant \sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} c(u,v)-\sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} f(v,u)$$

$$=c(V_1,\bar{V}_1)-\sum_{\substack{u\in V_1\\ v\in \overline{V_1}}} f(v,u)$$

$$\leqslant c(V_1,\bar{V}_1) \quad ∎$$

推论 2：网络的最大流小于等于最小切割的容量，即

$$\max(f(s,t))\leqslant \min(C(K)) \tag{9.6}$$

从(9.5)式可知，由于$f(s,t)$和$C(V_1,\bar{V}_1)$的任意性，(9.6)式显然成立。

§9.2 最大流最小切割定理

式(9.6)给出了任意一个网络最大流的上界，即网络的流不可能超过最小切割的容量。1956年福特(Ford)和富克逊(Fulkerson)提出网络的最大流恰好等于最小切割的容量，即把(9.6)式变为等式，称为最大流最小切割定理。在介绍和证明这一定理之前，先就需要用到的几个术语定义如下

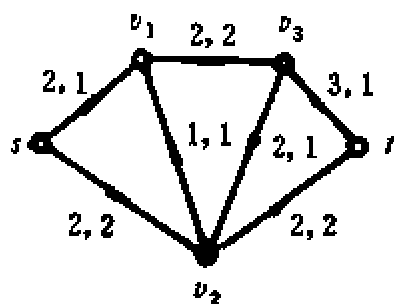


图 9.2

定义 9.10 网络$N=(V,E)$的一条从发点s到收点t的路Q是N的一个不同结点序列$Q=(v_0, v_1, \cdots, v_k)$，其中$v_0=s$, $v_k=t$，且Q的任意两个顺序结点v_i和v_{i+1}，或者$\langle v_i, v_{i+1}\rangle\in E$，或者$\langle v_{i+1}, v_i\rangle\in E$

如图9.2所示的网络，$Q_1=(s, v_1, v_2, v_3, t)$是$N$的一条从$s$到$t$的路，$Q_2=(s, v_2, v_1, v_3, t)$也是一条从$s$到$t$的路。

上面定义的路与第二章定义的有向路径不同之处在于，这里定义的路允许路上边的方向与路的走向相反。如果不考虑边的方向，则路的定义与无向图的通路的定义是一致的，也就是说，这里是对有向图的底图定义通路的。

定义 9.11 在网络 $N=(V, E)$ 的一条路 $Q=(s, v_1, v_2, \cdots, t)$ 中，若 $\langle v_i, v_{i+1}\rangle \in E$，则称此边为路的前向边，若 $\langle v_{i+1}, v_i\rangle \in E$，则称此为路的反向边。

如图 9.2，在路 Q_1 中 $\langle s, v_1\rangle$，$\langle v_3, t\rangle$ 是路的前向边，$\langle v_2, v_1\rangle$，$\langle v_3, v_2\rangle$ 是路的反向边。在 Q_2 中，$\langle v_2, v_1\rangle$ 则是路的前向边。由此可见，同一条边是前向边还是反向边，完全取决于该条路的走向。

定义 9.12 设 F 是网络 $N=(V, E)$ 上的流，$f(s, t)$ 为其流值，Q 是 N 的一条路，对路 Q 上的任一条边

（a）若 $\langle v_i, v_{i+1}\rangle$ 是前向边且

$$\Delta_i=C(v_i, v_{i+1})-f(v_i, v_{i+1})>0$$

并且

（b）若 $\langle v_{i+1}, v_i\rangle$ 是反向边且

$$\Delta_i=f(v_{i+1}, v_i)>0$$

则称 Q 为流的可增广路。否则称为流的不可增广路。

由定义可知，若 Q 的所有前向边都不是饱和边，并且所有的反向边的流都不为零，则 Q 是一条可增广路。

例如图 9.2 所示的路 Q_1，前向边 $\langle s, v_1\rangle$ 和 $\langle v_3, t_2\rangle$ 都不饱和，反向边 $\langle v_2, v_1\rangle$ 和 $\langle v_3, v_2\rangle$ 的流都不为零，故 Q_1 是可增广路。对于 Q_2，因前向边有饱和边，因而不是可增广路。

如果 Q 是一条可增广路，我们定义流的增量 Δ_0 如下

$$\Delta_0=\min\Delta_i>0$$

对于 Q 上的任一条边，如果它的 $\Delta_i=\Delta_0$，则称这条边为 Q 相对于流的瓶颈边。

例如图 9.2 中的可增广路 Q_1，对前向边 $\langle s, v_1\rangle$ 有

$$\Delta_i=C(s, v_1)-f(s, v_1)=2-1=1$$

对前向边 $\langle v_3, t\rangle$ 有

$$\Delta_3=C(v_3, t)-f(v_3, t)=3-1=2$$

对于反向边 $\langle v_2, v_1\rangle$ 和 $\langle v_3, v_2\rangle$，都有

$$\Delta_1=\Delta_2=1$$

故

$$\Delta_0=\min\{1, 2, 1, 1\}=1$$

此时，边 $\langle s, v_1\rangle$，$\langle v_2, v_1\rangle$ 和 $\langle v_3, v_2\rangle$ 都是路 Q 关于流的瓶颈边。

设 F 是当前网 N 上的流，如果存在可增广路 Q，我们就可以构造一新的网上流 F'，使其流值 $f'(s, t)$ 比原来的流值 $f(s, t)$ 增加一个增量 Δ_0，方法是对 Q 的每一条边按下述规则改变它的流量：

（a）对前向边 $\langle v_i, v_{i+1}\rangle$，则

$$f(v_i, v_{i+1})\leftarrow f(v_i, v_{i+i1})+\Delta_0$$

（b）对反向边 $\langle v_{i+1}, v_i\rangle$，则

$$f(v_{i+1}, v_i)\leftarrow f(v_{i+1}, v_i)-\Delta_0$$

显然这种改变对每个中间结点，(9.2)式仍然满足，而流经每条边的流仍然满足(9.1)式，所以 F' 仍是可行流，但其流值 $f'(s,t)$ 较前增加了一个增量 Δ_0。

例如对图 9.2 中的可增广路 Q_1，我们按上述法则改变各边的流，即

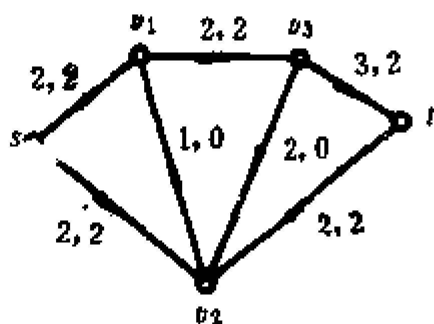


图 9.3

$$f(s,v_1)\leftarrow 1+1$$
$$f(v_2,v_1)\leftarrow 1-1$$
$$f(v_3,v_2)\leftarrow 1-1$$
$$f(v_3,t)\leftarrow 1+1$$

得到的 F' 如图 9.3 所示，此时 $f'(s,t)=4$，比原来的流值增加了 1。

定理 9.2 若网上的流 F 不存在可增广路，则其流值 $f(s,t)$ 即是网的最大流。

证：首先我们对网 N 的各结点进行标记，设各结点开始时均未标记，则结点的标记过程按如下步骤进行

（1）对发点 s 标记

（2）对于边 $\langle u,v\rangle\in E$，若 u 已标记而 v 未标记，并且有 $f(u,v)<C(u,v)$，则给 v 标记。

（3）对于边 $\langle u,v\rangle\in E$，若 v 已标记而 u 未标记，并且有 $f(u,v)>0$，则给 u 标记。

反复进行步骤(2)和(3)直到 N 的所有可能标记的结点都得到标记为止，可以看到，如果不存在可增广路，按照上述步骤收点 t 将不会得到标记，由标记过程我们定义一个切割 $(V_1,\bar{V}_1)$，即将所有已标记的结点划入子集 V_1 而将未标记的结点划入其补集 $\bar{V}_1$ 中，那么根据标记的规则我们可以得出：

如果 $u\in V_1$ 且 $v\in\bar{V}_1$，则 $f(u,v)=C(u,v)$

如果 $u\in\bar{V}_1$ 且 $v\in V_1$，则 $f(u,v)=0$

于是由定理 9.1 即得

$$f(s,t)=\sum_{\substack{u\in V_1\\ v\in\bar{V}_1}}f(u,v)-\sum_{\substack{u\in V_1\\ v\in\bar{V}_1}}f(v,u)=\sum_{\substack{u\in V_1\\ v\in\bar{V}_1}}c(u,v)=c(V_1,\bar{V}_1)$$

由推论 2 可知此时的 $f(s,t)$ 一定是 N 上流的最大值。 ▮

通过对上面定理的证明，即可得出最大流最小切割定理如下

定理 9.3 在一个给定的网络 N 中，流的最大值等于切割的最小容量，即

$$\max(f(s,t))=\min(C(K)) \tag{9.7}$$

证：由定理 9.2 已知，当不存在可增广路时，网上的流值即为最大流。我们还可以肯定，按标记过程定义的切割 $(V_1,\bar{V}_1)$ 是容量最小的切割，否则如果还有容量更小的切割 $(V_1',\bar{V}_1')$，将得出 $f(s,t)>C(V_1',\bar{V}')$，与推论 1 相矛盾，故

$$\max(f(s,t))=\min(C(K))$$ ▮

定理 9.2 的证明过程，为我们提供了寻找网的最大流的方法，即从网的初始流 F_0 开始，找出一条可增广路 Q_0，然后使这条路上的流值增加 Δ_0，网上的流变为 F_1，从中又找出一条可增广路 Q_1，又使路上的流增加 Δ_0'，网上的流成为 F_2，如此继续下去，每找到一条增广路就可使网的流值增加 Δ_0，推论 1 给出了流值的上界，说明这种过程不会是无止

境的，因此最终必然出现不再有任何可增广路，此时网上的流值 $f(s, t)$ 即是最大流。

怎样寻找一条可增广路，定理的证明也为我们提供了一种方法，即下节要介绍的标记法。

§9.3 标记法

这一算法分两个过程，一是标记过程，二是增广过程；前一过程通过对结点的标记寻找一条可增广路，后一过程则使沿可增广路的流增加。

在标记过程中，每一结点给予三个标号，第一个标号表示该点的先驱点，第二个标号为"+"或"−"表示先驱点与该点连接的边在可增广路中是前向边还是反向边，第三个标号表示这条边上能增加（或减少）的流值 Δ_i

一、标记过程 A

A_1（第一步）：发点 S 标记为（S，+，$\Delta(S)=\infty$）此时 S 称为已标记、未检查，其余的点均称为未标记、未检查。

A_2（第二步）：任选一已标记未检查的结点 u，若结点 v 与 u 邻接且未标记，则当：

（1）若 $\langle u, v\rangle \in E$ 且 $C(u, v)>f(u, v)$ 时，则将 v 标记为（u，+，$\Delta(v)$），其中

$$\Delta(v)=\min\{\Delta(u),\ C(u, v)-f(u, v)\}$$

之后，称 v 已标记、未检查。

（2）若 $\langle v, u\rangle \in E$ 且 $f(v, u)>0$ 时，则将 v 标记为（u，−，$\Delta(v)$），其中

$$\Delta(v)=\min\{\Delta(u),\ f(v, u)\}$$

之后，称 v 已标记、未检查。

（3）与结点 u 邻接的所有结点都标记完之后，将 u 的标记中的符号"+"或"−"加上小圆圈，表示 u 已标记且已检查。

A_3（第三步）：重复步骤 A_2，直到收点 t 被标记，或者收点不可能获得标记为止。如果是前者，转向增广过程，如果是后者，算法结束，所得流即是最大流。

二、增广过程 B

B_1（第一步）：令 $z=t$

B_2（第二步）：若 z 的标记为（q，+，$\Delta(z)$），则

$$f(q, z)\longleftarrow f(q, z)+\Delta(t)$$

若 z 的标记为（q，−，$\Delta(z)$），则

$$f(z, q)\longleftarrow f(z, q)-\Delta(t)$$

B_3（第三步）：如果 $q=s$，则把全部标记去掉，转向标记过程 A，否则令 $z=q$，转到 B_2。

为了证明这一算法，特举一例如下

例 9.2 用标记法求图 9.4 所示网络的最大流（图中边上的权表示容量）

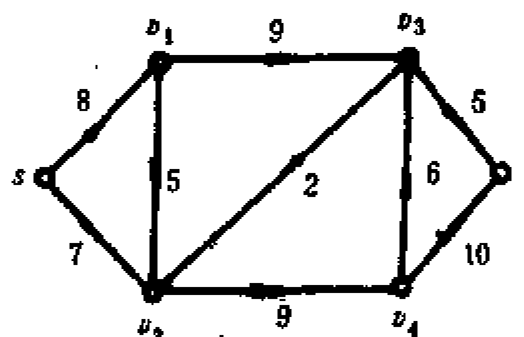


图 9.4

解：F_0 可任意假定一初始值，但必须是可行流，即满足式（9.1）和（9.2）。这里我

们从零流开始，设网上每条边的流均为零，即 $f(u, v)=0$，$\langle u, v\rangle \in E$

（一）找一条可增广路并增加其流值

A（标记过程）：

（1）发点 S 标记为（S，+，∞）

（2）考察与 S 邻接的点 v_1 和 v_2（为了简单起见，以后用点的下标表示该点）

对点 1，因 $\langle S, 1\rangle \in E$，且 $C(S, 1)>f(S, 1)$，故

$$\Delta(1)=\min\{\Delta(S), C(S, 1)-f(S, 1)\}=8$$

于是点 1 标记为（S，+，8）

对点 2，用同样方法得到标记为（S，+，7）

至此，与 S 邻接的结点都已标记，S 标记中的“+”写成“⊕”，表示 S 已标记、已检查，如图 9.5 所示。

（3）重复步骤 A_2，选一已标记、未检查的点，例如选择点 1，与 1 邻接且未标记的点只有点 3，因 $\langle 1, 3\rangle \in E$，且 $C(1, 3)>f(1, 3)$，故

$$\Delta(3)=\min\{\Delta(1), C(1, 3)-f(1, 3)\}=8$$

于是点 3 标记为（1，+，8）

至此，点 1 已标记已检查，将它标记中的“+”圈上小圈，如图 9.6 所示

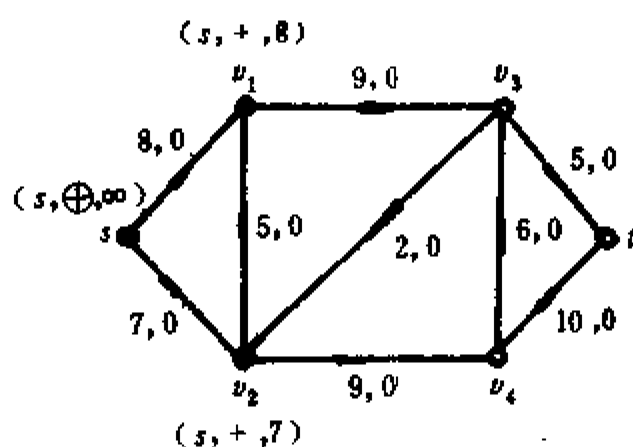


图 9.5

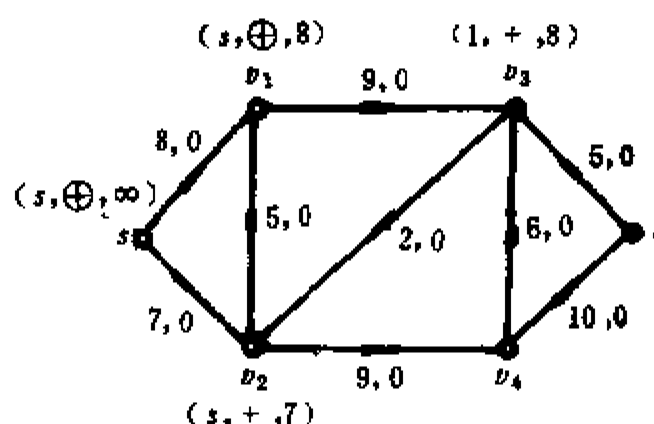


图 9.6

（4）重复步骤 A_2，选一已标记未检查的点，例如选择点 3，与它邻接且未标记的点有 4 和 t。

对于点 4，因 $\langle 4, 3\rangle \in E$ 且 $f(4, 3)=0$，因此不能用点 3 去标记点 4。

对于点 t，因 $\langle 3, t\rangle \in E$ 且 $C(3, t)>f(3, t)$，故

$$\Delta(t)=\min\{\Delta(3), C(3, t)-f(3, t)\}=5$$

于是 t 标记为（3，+，5），如图 9.7。由于 t 已被标记，转到增广过程 B

B（增广过程）

$$f(3, t)\longleftarrow 0+5=5$$

$$f(1, 3)\longleftarrow 0+5=5$$

$$f(s, 1)\longleftarrow 0+5=5$$

至此完成一次增广过程，如图 9.8 所示。

（二）找一条可增广路并增加其流值

A（标记过程）：对图 9.8 重新标记，得到图 9.9。

B（增广过程）：从标记过程得到一条可增广路 $S\to v_2\to v_4\to t$，增值 $\Delta_0=7$，于是得

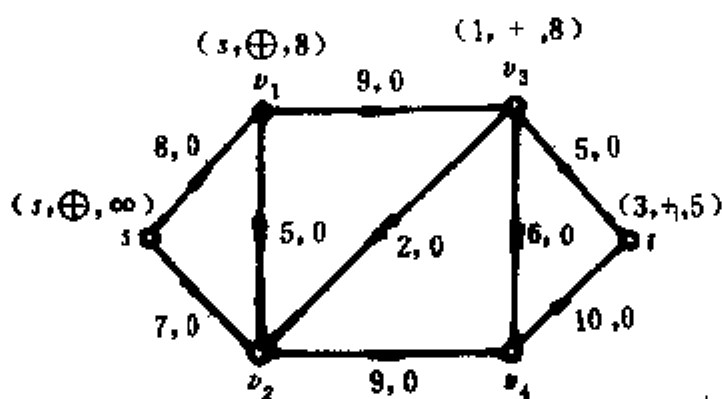


图 9.7

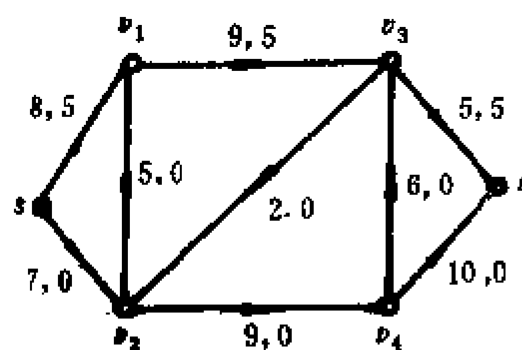


图 9.8

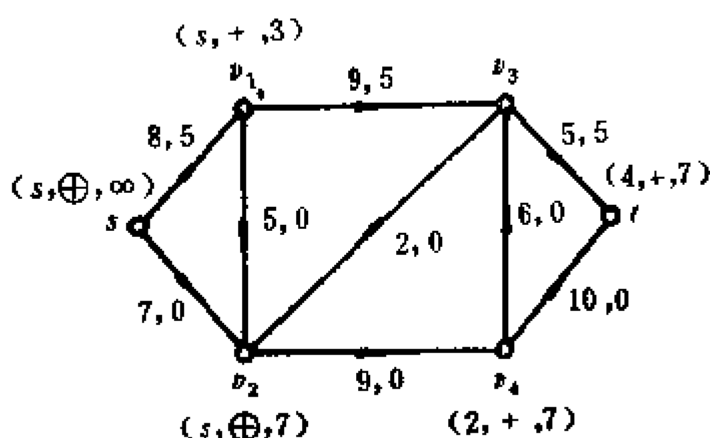


图 9.9

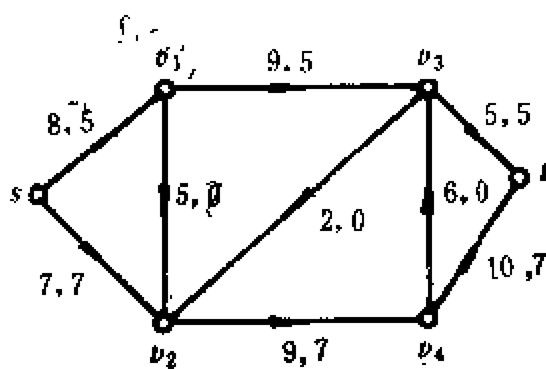


图 9.10

到图 9.10，至此又完成一次增广过程。

（三）找一条可增广路并增加其流值

对图 9.10 重新标记，得到图 9.11，从中找到一条增广路 $S\to v_1\to v_2\to v_4\to t$，增值为 2，于是得到图 9.12。

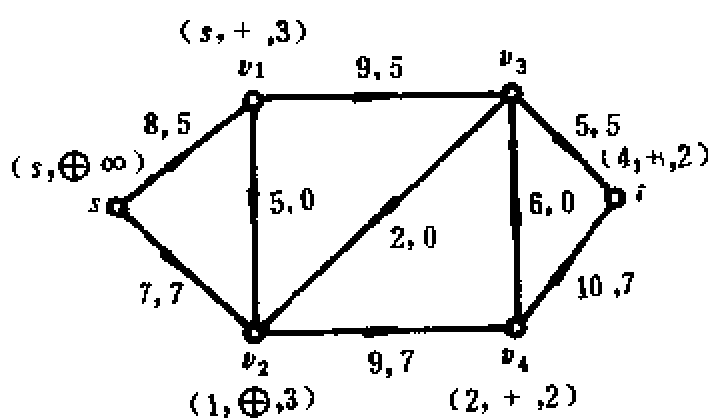


图 9.11

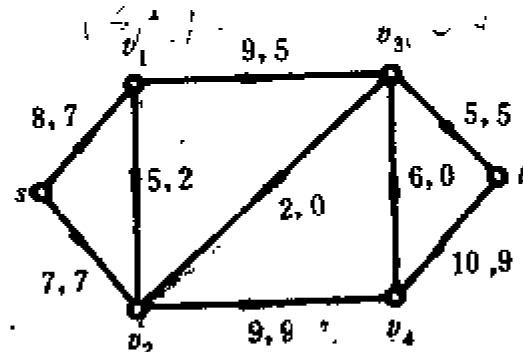


图 9.12

（四）对图 9.12 重新标记，得到图 9.13，可见 v_4 和 t 都不可能再获得标记，算法结束。网上最大流为

$$\max f(s,\ t)=14$$

将获得标记的结点归入 V_1，不能标记的结点归入 $\bar{V}_1$，即 $V_1=\{s,\ v_1,\ v_2,\ v_3\}$，$\bar{V}_2=\{v_4,\ t\}$，得到最小切割为

$$(V_1,\ \bar{V}_1)=\{(v_2,\ v_4),\ (v_3,\ t)\}$$

其容量为

$$C(V_1,\ \bar{V}_2)=9+5=14$$

可见网的最大流等于最小切割的容量。

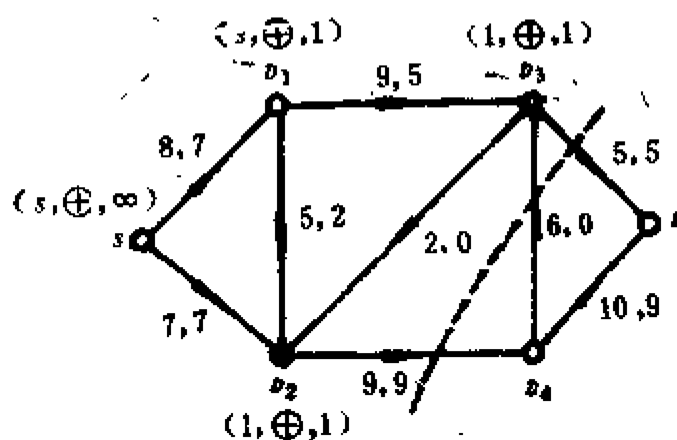


图 9.13

最后必须指出一点，在标记过程中，如果选取标记点的次序不同，可能会得出不同的可增广路，因而网上流的分配可能不一样，但最大流的值是相同的。

§9.4 最短路径法

上一节讲的标记法有一明显不足之处，就是如果每一次找到的可增广路只能增加一个单位的流量时，那么从零流开始计算需要进行增广过程的迭代次数将等于网络最小切割的容量 $\min C(V_1, \bar{V}_1)$，这一数值可以是任意大的数并且与网络的大小（即网络的结点和边数的多少）无关。例如图 9.14 表示一个最简单的网络，设边 $\langle v_1, v_2\rangle$ 的容量是 1，其余各条边的容量均为K，从零流开始，交替地采用可增广路 $P_1=(s, v_1, v_2, t)$ 和 $P_2=(s, v_2, v_1, t)$，每一次增广只增加 1 个单位的流量，于是需要反复进行 $2K$ 次。

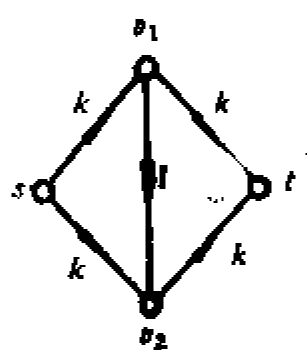


图 9.14

一般都希望一个算法，它的时间复杂性仅取决于网络的规模，并且是低阶多项式的。为此不少人提出了各种改进的算法，下面我们介绍由Edmonds和Karp（1972）提出的一个算法，它的时间复杂性为 $O(n|E|^2)$ 级的。

算法的基本思想仍然是通过寻找可增广路增加网络的流，最后达到最大流。但是与上面标记法不同，这里是根据最短路径来选择可增广路，因而可以避免上面那种现象的发生，使迭代次数减少。

设网络 $N=(V, E)$，网上的初始流为F，根据网上流F，我们构造一个网络 $N^F=(V, E')$，这里N和N^F有着相同的结点集合，E'是这样确定的，即对任意两个结点 u 和 v，$\langle u, v\rangle\in E'$ 当且仅当：

$$\langle u, v\rangle\in E \text{ 并且 } C(u, v)-f(u, v)>0$$

或
$$\langle v, u\rangle\in E \text{ 并且 } f(v, u)>0 \qquad (9.8)$$

例 9.3 如图 9.15 所示的网络 $N=(V, E)$，边的容量及流量已标出，求 N^F

解：因 $\langle s, v_1\rangle\in E$，且 $C(s, v_1)>f(s, v_1)$，故有 $\langle s, v_1\rangle\in E'$

另一方面，因 $\langle s, v_1\rangle\in E$ 且 $f(s, v_1)>0$，故 $\langle v_1, s\rangle\in E'$

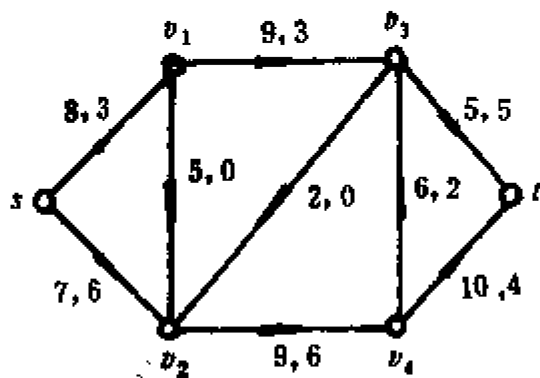


图 9.15

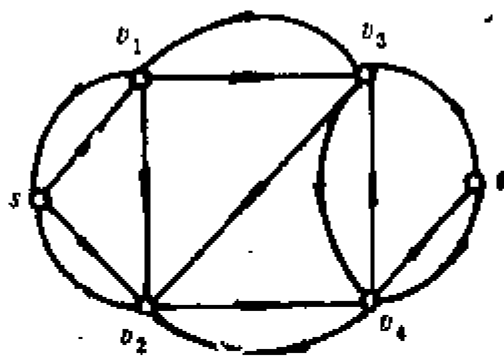


图 9.16

同理有 $\langle v_1, v_3\rangle\in E'$ 及 $\langle v_3, v_1\rangle\in E'$。

此外，虽然 $\langle v_3, t\rangle\in E$，但 $C(v_3, t)=f(v_3, t)$，故 $\langle v_3, t\rangle\notin E'$。

同理，虽然 $\langle v_3, v_2\rangle\in E$，但 $f(v_3, v_2)=0$，故 $\langle v_2, v_3\rangle\notin E'$

其他情况类似可求，于是得到 N^F 如图 9.16 所示。

这样，求网络N上从s到t的可增广路就可化归为求网络N^F上的从s到t的一条有向路径，设N上一条可增广路为P，在N^F上与P对应的一条有向路径为P^F，显然它们有着一一对应的关系，而且N中的边如果是瓶颈边，在N^F中与之对应的边亦是瓶颈边。

本算法的基本思路就是首先在N^F中找一条从s到t的最短有向路径，然后在N中得到对应的一条可增广路（路径长度是指路径中边的数目，与边的权无关），算法中寻找最短路径作为一个过程在主程序中调用。算法如下。

一、求最短路径P^K过程，BFSPK

begin

1．用广度优先搜索法求出N^F中从s到其余各点v的距离$L(v)$（$v \neq s$，如果$L(v)=0$，表示从s到v不存在通路）

2．if $L(t)=0$ then PATH⟵false

else

begin

3．for 所有的$v \in V$，求出$B'(v)$

4．P^F⟵(t)

5．u⟵t

6．while $u \neq s$ do

begin

7．找一点v，$v \in B'(u)$且$L(u)=L(v)+1$

8．将v加到P^F的首端

9．u⟵v

end

end

end

二、求最大流算法

1．输入各结点的邻接表$A(v)$，边的容量及边中的初始流。

2．PATH⟵true

3．while PATH=true do

begin

4．求出N^K中各结点的邻接表$B(v)$，对每一条边$\langle u, v\rangle$，不管是前向边或是反向边，记$F\Delta(u, v)$

5．BFSPK

6．if PATH=true then

begin

7．求出$\Delta_0=\min\Delta(u, v)$，$\langle u, v\rangle \in P$

8．for 所有$\langle u, v\rangle \in P$ do

9．if $\langle u, v\rangle$是P的前向边

10. then $f(u, v) \longleftarrow f(u, v)+\Delta_0$

11. else $f(v, u) \longleftarrow f(v, u)-\Delta_0$

  end

end

三、算法说明

1. 在求最短路径 P^K 过程（$BFSPK$）中，实际上包含了用广度优先搜索法求从给定点 s 到其余各点的距离的过程，（即第 1 行），如果在 N^F 中存在从 s 到 t 的有向路径，则选一条长度最小的路径作为 P^K，方法是从 t 逆向寻迹，即如果 v 是当前访问的结点，则下一个访问点 u 应有 $L(u)=L(v)-1$。

2. 对 $BFSPK$ 的第 3 行，$B'(v)$ 为 N^F 中所有引入 v 的边的起点集合，即

$$B'(v)=\{u \mid \langle u, v\rangle \in E'\}$$

3. 对主程序的第 2 行，设置一布尔变量 PATH，其初始值为 true，当 PATH 为 false 时，即 $L(t)=0$，表示已不存在从 s 到 t 的通路，亦不存在从 s 到 t 的可增广路，此时网上的流已达最大值，算法结束。

4. 主程序的第 4 行，$B(v)$ 是在网络 N^F 中，结点 v 的邻接表，即满足（9.8）式的所有与 v 邻接的结点集合。

5. 主程序的第 7 行～第 11 行是对网络 N 的增广过程，至此完成一次迭代。从第 3 行～第 11 行表示增广的迭代过程，每完成一次迭代过程，就重新构造网络 N^K（第 4 行）进行下一次增广过程。

例 9.4 试用最短路径法求图 9.17 所示网络 $N=(V, E)$ 的最大流，边的容量及初始流已标在图上

解

（一）1. 构造 N'（见例 9.3）得到图 9.18。

2. 求出各结点的 $B(v)$ 如下（在网络 N^F 中 $B(v)$ 即为从 v 引出的边的终止点集合）：

$$B(s)=(v_1, v_2)$$

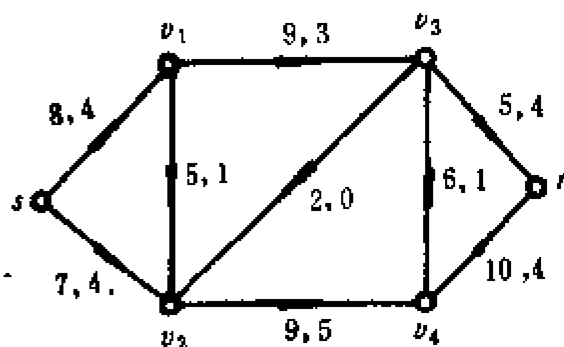


图 9.17

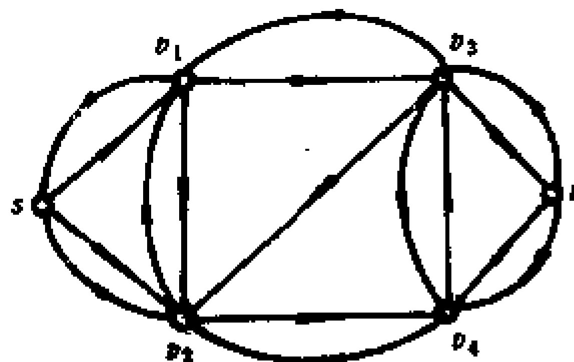


图 9.18

$B(v_1)=(s, v_2, v_3)$，　　$B(v_2)=(s, v_1, v_4)$

$B(v_3)=(v_1, v_2, v_4, t)$，$B(v_4)=(v_2, v_3, t)$

$B(t)=(v_3, v_4)$

3. 求 $L(v)$（s 到 v 的最短路径长度）

$L(s)=0$，$L(v_1)=1$，$L(v_2)=1$，$L(v_3)=2$，$L(v_4)=2$，$L(t)=3$

4．求$B'(v)$（在N^F中引入v的边的始点集）

$B'(s)=(v_1, v_2)$，$B'(v_1)=(s, v_2, v_3)$，$B'(v_2)=(s, v_1, v_3, v_4)$，$B'(v_3)=(v_1, v_4, t)$，$B'(v_4)=(v_2, v_3, t)$，$B'(t)=(v_3, v_4)$

5．$P^F \leftarrow (t)$，$u \leftarrow t$，在$B'(t)$中找到点v_4满足$L(t)=L(v_4)+1$，故$P^F=(v_4, t)$

$u \leftarrow v_4$，在$B'(v_4)$中，找到点v_2符合条件，故$P^F=(v_2, v_4, t)$

$u \leftarrow v_2$，在$B'(v_2)$中，找到点s符合条件，故$P^F=(s, v_2, v_4, t)$

至此在N^F中找到一条从s到t的最短路P^F，它对应着N中一条可增广路P

6．求$\Delta_0=\min\Delta(u, v)$，$(u, v)\in P$

$$\Delta_0=\min\{\Delta(s, v_2), \Delta(v_2, v_4), \Delta(v_4, t)\}$$
$$=\min\{(7-4), (9-5), (10-4)\}=3$$

7．$f(s, v_2)=4+3=7$

$f(v_2, v_4)=5+3=8$

$f(v_4, t)=4+3=7$

至此，完成第一次流的增广，得到图9.19。

（二）1．构造N^2，得到图9.20。

2．求出$B(v)$，$L(v)$，$B'(v)$如下：

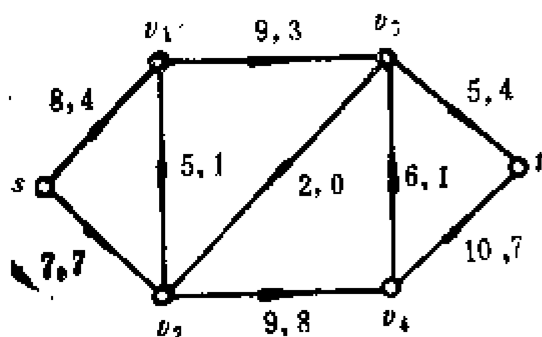


图 9.19

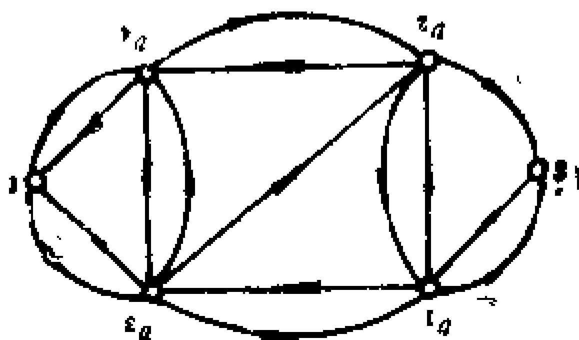


图 9.20

$B(s)=(v_1)$，$B(v_1)=(s, v_2, v_3)$，$B(v_2)=(s, v_1, v_4)$，$B(v_3)=(v_1, v_2, v_4, t)$，$B(v_4)=(v_2, v_3, t)$，$B(t)=(v_3, v_4)$

$L(s)=0$，$L(v_1)=1$，$L(v_2)=2$，$L(v_3)=2$，$L(v_4)=3$，$L(t)=3$，

$B'(s)=(v_1, v_2)$，$B'(v_1)=(s, v_2, v_3)$，$B'(v_2)=(v_1, v_3, v_4)$，$B'(v_3)=(v_1, v_4, t)$，$B'(v_4)=(v_2, v_3, t)$，$B'(t)=(v_3, v_4)$

3．求P^F。

$P^F \leftarrow (t)$，在$B'(t)$中符合条件的只有v_3，在$B'(v_3)$中符合条件的只有v_1，故得到

$$P^F=(s, v_1, v_3, t)$$

4．求Δ_0

$\Delta_0=\min\{\Delta(s, v_1), \Delta(v_1, v_3), \Delta(v_3, t)\}=1$

5．$f(s, v_1)=4+1=5$，$f(v_1, v_3)=3+1=4$，$f(v_3, t)=4+1=5$

至此，完成流的第二次增广，得到图9.21

（三）1．构造N^3，得到图9.22

2．求$B(v)$，$L(v)$，$B'(v)$：

$B(s)=(v_1)$，$B(v_1)=(s, v_2, v_3)$，$B(v_2)=(s, v_1, v_4)$，$B(v_3)=(v_1, v_2, v_4)$，$B(v_4)=$

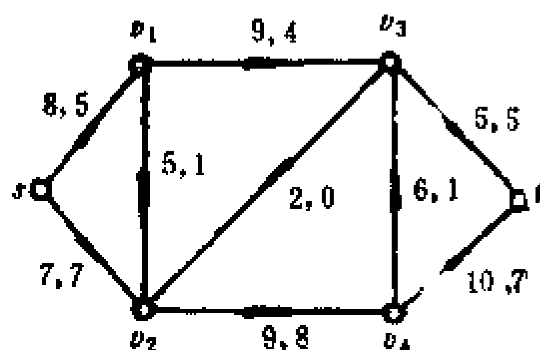


图 9.21

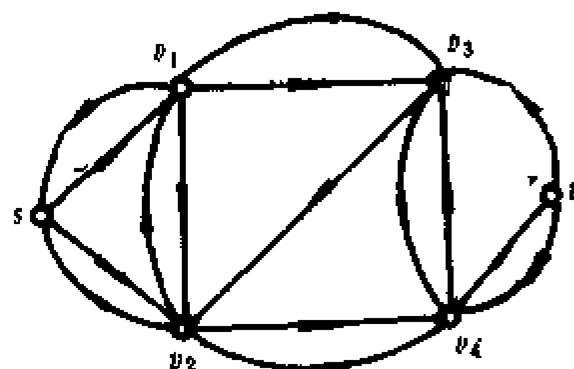


图 9.22

(v_2, v_3, t), $B(t)=(v_3, v_4)$

$L(s)=0$, $L(v_1)=1$, $L(v_2)=2$, $L(v_3)=2$, $L(v_4)=3$, $L(t)=4$

$B'(s)=(v_1, v_2)$, $B'(v_1)=(s, v_2, v_3)$, $B'(v_2)=(v_1, v_3, v_4)$, $B'(v_3)=(v_1, v_4, t)$, $B'(v_4)=(v_2, v_3, t)$, $B'(t)=(v_4)$

3. $P^F \leftarrow (t)$，在 $B'(t)$ 中符合条件的只有 v_4，在 $B'(v_4)$ 中符合条件的有 v_2 和 v_3，设选取 v_2，在 $B'(v_2)$ 中符合条件的只有 v_1，于是

$$P^F=(s, v_1, v_2, v_4, t)$$

4. $\Delta_0=\min\{3, 4, 1, 3\}=1$

5. 对路径 P 中各边的流量均增加 1，至此完成流的第 3 次增广，得到图 9.23

(四) 1. 构造 N^4，得到图 9.24

2. 求 $B(v)$，$L(v)$，$B'(v)$

$B(s)=(v_1)$, $B(v_1)=(s, v_2, v_3)$, $B(v_2)=(s, v_1)$, $B(v_3)=(v_1, v_2, v_4)$, $B(v_4)=(v_2, v_3, t)$, $B(t)=(v_3, v_4)$

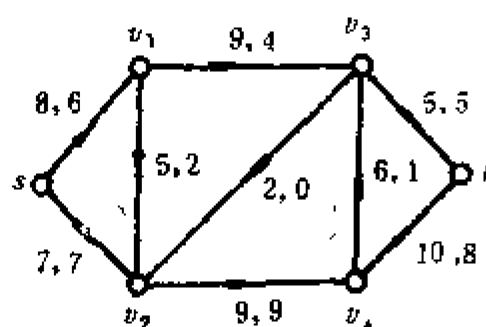


图 9.23

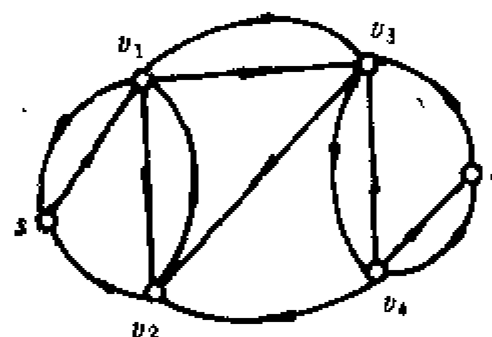


图 9.24

$L(s)=0$, $L(v_1)=1$, $L(v_2)=2$, $L(v_3)=2$, $L(v_4)=3$, $L(t)=4$

$B'(s)=(v_1, v_2)$, $B'(v_1)=(s, v_2, v_3)$, $B'(v_2)=(v_1, v_3, v_4)$, $B'(v_3)=(v_1, v_4, t)$, $B'(v_4)=(v_3, t)$, $B'(t)=(v_4)$

3. $P^F \leftarrow (t)$，$u \leftarrow t$，在 $B'(t)$ 中符合条件的只有 v_4，在 $B'(v_4)$ 中符合条件的只有 v_3，在 $B'(v_3)$ 中符合条件的只有 v_1，于是

$$P^F=(s, v_1, v_3, v_4, t)$$

4. $\Delta_0=\min\{2, 5, 1, 2\}=1$

5. 对边 $\langle s, v_1\rangle$，$\langle v_1, v_3\rangle$，$\langle v_4, t\rangle$ 中的流量增加 1，对反向边 $\langle v_4, v_3\rangle$ 的流量减 1，至此完成流的第 4 次增广，得到图 9.25

(五) 1. 构造 N^5 得到图 9.26

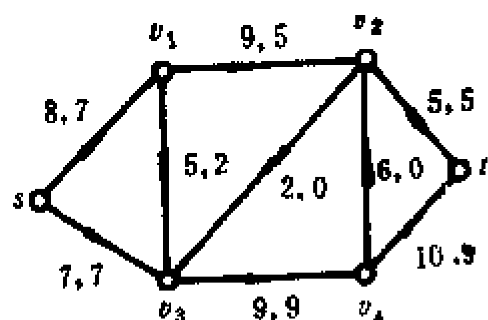


图 9.25

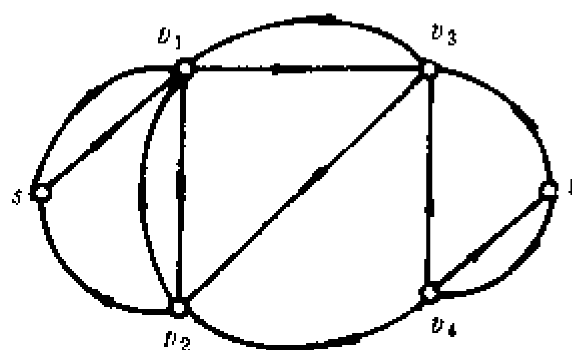


图 9.26

2. $B(s)=(v_1)$, $B(v_1)=(s, v_2, v_3)$, $B(v_2)=(s, v_1)$, $B(v_3)=(v_1, v_2)$, $B(v_4)=(v_2, v_3, t)$, $B(t)=(v_3, v_4)$

$L(s)=0$, $L(v_1)=1$, $L(v_2)=2$, $L(v_3)=2$, $L(v_4)=0$, $L(t)=0$

因 $L(t)=0$, PATH←false, 计算结束。

可以证明[1]，用最短路径法求网络的最大流，其迭代次数不会超过$\frac{1}{2}|E|\cdot n$，从算法中可以看到，每一次迭代其计算复杂性都是$O(|E|)$级的，因而总的计算量为$O(n|E|^2)$。虽然已经出现一些求最大流的算法，其计算量可以降低到$O(n^3)$级，但由于这些算法要求较多的注释，我们就不一一介绍了。

§9.5 最大流最小切割定理的应用推广

在许多看起来与最大流问题不同的问题，如果稍加修改可化为最大流问题求解，这一节我们介绍有关这方面的几个例子。

一、多产地多销地问题

如果运输网络不止一个发点和一个收点，而是有多个发点 s_1, s_2, …, s_m 和多个收点 t_1, t_2, …, t_n，假设都是输送同一种货物，对此，我们可以在网络中加入一个假想发点 s 和一个假想收点 t，并增添新的有向边 $\langle s, s_1\rangle$, $\langle s, s_2\rangle$, …, $\langle s, s_m\rangle$ 和 $\langle t_1, t\rangle$, $\langle t_2, t\rangle$, …, $\langle t_n, t\rangle$，同时指定这些边的容量一律为$+\infty$，如图 9.27 所示

这样多发点多收点的网络就化归为单发点单收点的网络了，用上一节的算法求出最大流后，把通过 s_i 点的流作为发点 s_i 发出的流，而把经由 t_j 的流作为流入收点 t_j 的流。

例 9.5 如图 9.28 所示的双发点双收点的运输网络，求最大流和最小切割。

解：增加一虚拟发点 s 和一虚拟收点 t 及边 $\langle s, s_1\rangle$, $\langle s, s_2\rangle$, $\langle t_1, t\rangle$, $\langle t_2, t\rangle$ 并令其容量均为∞，如图 9.29 所示，采用上一节算法即得

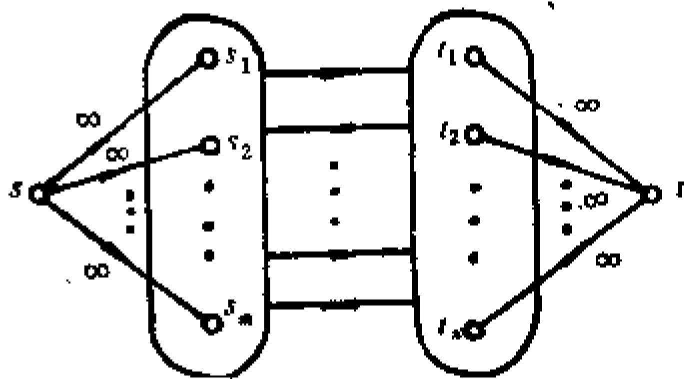


图 9.27

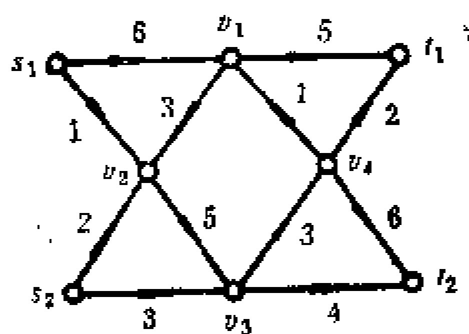


图 9.28

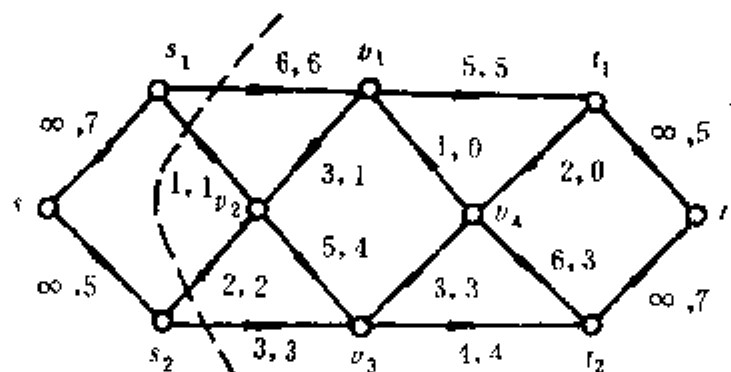


图 9.29

$\max f(s,\ t)=12,\ \min(V,\ \bar{V}_1)=\{(s_1,\ v_1),\ (s_1,\ v_2),\ (s_2,\ v_1),\ (s_2,\ v_3)\}$

二、点权网络的极大流

假如边的容量不受限制，只有结点的容量受到限制，即为点权网络。对于点权网络可以化成前面讨论的边权网络进行计算。设结点 v_i 的容量为$C(v_i)$，计算步骤如下：

（1）将 v_i 分为两个结点 v'_i 和 v''_i

（2）在 v'_i 与 v''_i 之间连一条有向边 (v'_i, v''_i)，使 $C(v_i', v_i'')=C(v_i)$

（3）原来网络中，所有引向 v_i 的边均使引向v_i'，所有由 v_i 引出的边均改由 v_i''引出

（4）原来边的容量均令其为$+\infty$（如果原来边的容量是有限的话，仍可保持其数值）

对原网络的所有点按上面步骤归化之后，点权网络的最大流问题就化成为边权网络的最大流问题了，而且对点带权边也带权的综合网络，亦可用上面的方法求最大流。

例 9.6 求图 9.30 所示的综合网络的最大流。

解：将结点分解后得到图 9.31，用求边权网络最大流方法即得

$$\max f(s,\ t)=18$$

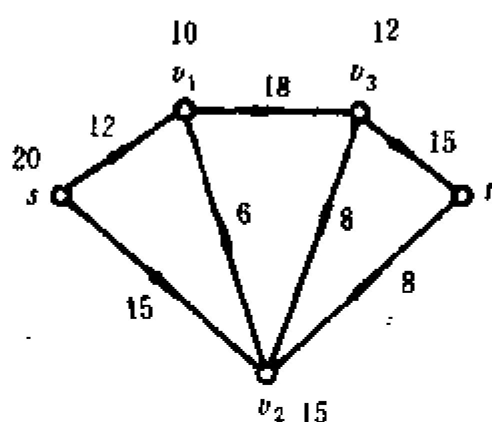


图 9.30

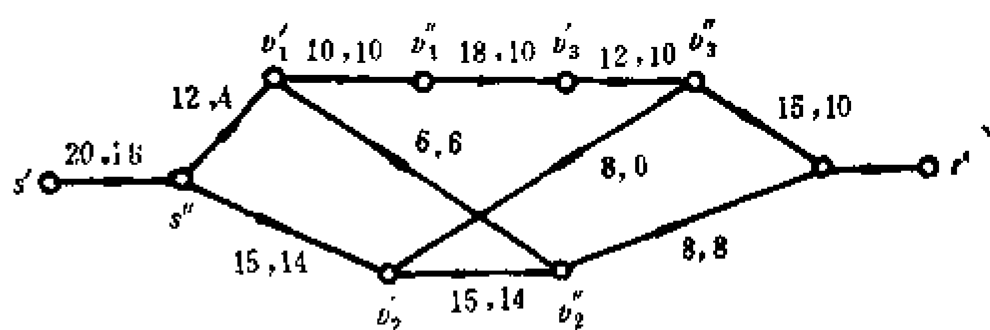


图 9.31

三、偶图的最大匹配

第五章讨论的偶图的最大匹配问题，实际上也可化为最大流问题来解决，它与多发点多收点网络的最大流算法类似，也是增添一个虚拟发点和收点，只是各条边的容量均设置为1，现举一例说明如下

例 9.7 求图 9.32 所示的偶图 $G=(V,E)$ 的最大匹配

解：由图G按下述步骤构造网络G'

（1）给G的所有边指定方向（从V_1指向V_2）

（2）增加一个发点s并从s到V_1的各结点引一有向边

（3）增加一个收点t并从V_2的各结点引一有向边到t

（4）令每一条边$\langle u,v\rangle$的容量$C(u,v)=1$

得到的网络G'如图9.33所示。

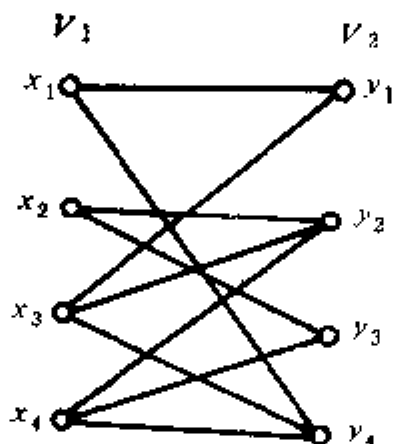


图 9.32

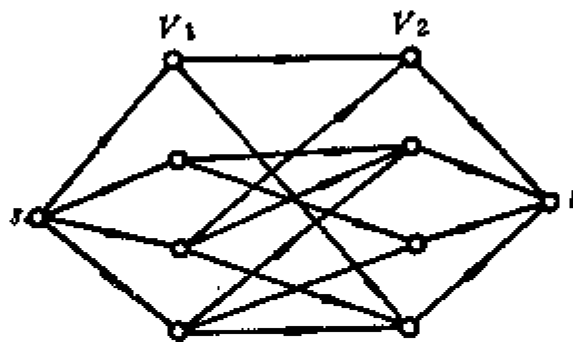


图 9.33

按求G'最大流的方法，凡流量为1的边即为匹配边。（与s或t关联的边除外）

§9.6 最小费用流

前面我们讨论网络上的流，只着眼于在可行流的条件下，怎样安排网上的流，使从发点输送到收点的流量达到最大。这一节我们要进一步探讨不仅要使网上的流达到最大，或者达到要求的预定值，而且还要使运输流的费用是最小的，这就是最小费用流问题。显然这一问题更具有实际意义，因为任何一个运输网络，不仅要考虑如何发挥网络的运输能力，而且还应该考虑如何降低运输的成本。因此，网络的任何一条边都有两个参数，一个是允许通过流量的最大值，即边的容量，另一个则是流过单位流量时需要的费用，一般来说，由于结构不同，网上不同边的容量不是完全相同的，同样道理，不同的边流过单位流量的费用也不一样。

定义 9.13 设F为网络$N=(V,E)$上的流，$a(u,v)$表示流过边$\langle u,v\rangle\in E$的单位流量的费用，则称

$$a(G,F)=\sum_{\langle u,v\rangle\in E}a(u,v)\cdot f(u,v) \tag{9.9}$$

为网N上通过流F的费用

求最小费用流也有多种算法，但都有一个共同之处，即在求最大流的算法上，进行一些修改或补充，使得到的解既是网上的最大流（或预定值），且总的费用$a(G,F)$是最小的。下面我们介绍两种应用较广的算法。

一、最短路径法

这一算法的基本思路是：先找一条从发点到收点单位流量费用最小的边构成的可增广

路，然后增加这条路的流到最大可能值，如此反复进行，直到网上的流达到最大值，此时总的费用也必然是网上相同流中的最小值。如果我们把边 $\langle u,v\rangle$ 的 $a(u,v)$ 作为边的长度，则寻找从发点到收点单位流量费用最小的边构成的路径，就变成了寻找从发点到收点的一条最短路径。因此得出如下算法

1．以 $a(u, v)$为边 $\langle u, v\rangle$ 的长度，求 s 到 t 单位流量的最短路径。

2．增加该路径的流使达到饱和，将路径中边的容量减去最大流作为边的新容量，并对前向饱和边，反向边和反向零流边的 $a(u, v)$ 作修改，得一新的网络

3．对新网络重复步骤（1），（2），直到再也找不到 s 到 t 的最短路，算法结束。

算法说明：

1．本算法应从零流开始。

2．在迭代过程中，边的长度（权）可能会出现负值，因此，这里将遇到求带负权的最短路径问题。

3．在执行第 2 步构造新的网络时，以及在寻找新网络中从 s 到 t 的最短路时，应遵守以下规则：

（a）当前向边是非饱和边时，即 $f(u, v)<C(u, v)$，令

$$w(u, v)=a(u, v)$$

$$c(u, v)\leftarrow c(u, v)-f(u, v)$$

（b）当前向边是饱和边时，即 $f(u, v)=c(u, v)$，令

$$w(u,v)=\infty, \qquad c(u,v)=0$$

（c）当反向边的流大于零时，即 $f(u,v)>0$，令

$$w(v, u)=-a(u, v)$$

$$c(u, v)\leftarrow c(u, v)+f(v,u)$$

（d）当反向边的流等于零时，即 $f(u, v)=0$，令

$$w(v,u)=\infty$$

（e）当反向边是饱和边时，令

$$w(v, u)=-a(u, v)$$

$$c(v, u)=c(u,v)$$

这里 $w(u,v)$表示边 $\langle u, v\rangle$ 的长度（权）如果 $\langle u, v\rangle$ 是反向边，则 $c(v, u)$表示沿路径方向这条边可以通过流的上限，$w(v, u)$则表示这条边的权（为负值）。

例 9.8 用最短路径法求图 9.34 的网络的最小费用流。图中边 $\langle u, v\rangle$ 上的数字表示 $a(u, v)$和 $c(u, v)$

解： 以 $a(u,v)$作为边 $\langle u,v\rangle$ 的长度 $w(u, v)$得到从 s 到 t 的一条最短路径为

$$P: s\longrightarrow a\longrightarrow b\longrightarrow t$$

其长度

$$L(p)=1+1+1=3$$

在这条路径上使流量增加到 2（最大可能值），于是边 $\langle s, a\rangle$，$\langle a, b\rangle$，$\langle b,t\rangle$ 均成为饱和边，得到一个新的网络如图 9.35 所示

求新网络从 s 到 t 的最短路径，有两条从 s 到 t 的通路，一条是

$$P_1: s\longrightarrow b\longrightarrow c\longrightarrow t$$

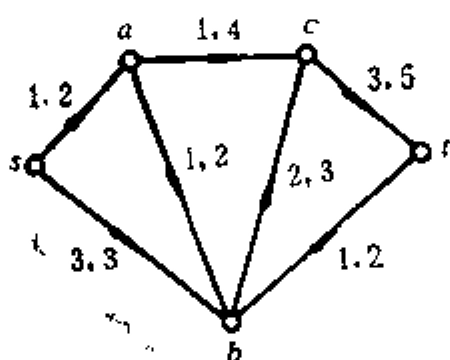


图 9.34

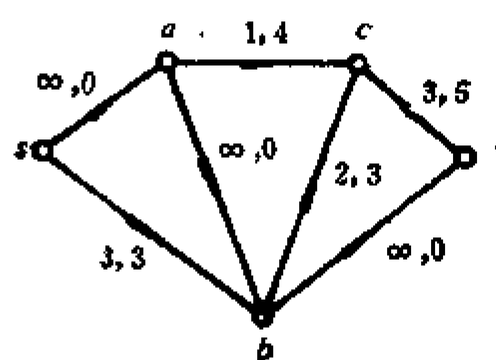


图 9.35

这条路径上的边都是前向边，其长度为

$$L(P_1)=3+2+3=8$$

另一条是

$$P_2:\ s\longrightarrow b\longrightarrow a\longrightarrow c\longrightarrow t$$

在这条路径上边 $\langle a, b\rangle$ 是反向边又是饱和边，故 $w(b, a)=-a(a, b)=-1$，其长度为

$$L(P_2)=3+(-1)+1+3=6$$

因此 P_2 是从 s 到 t 的最短路，这条路的流量增加到 2（$=c(a, b)$），由此得到新的网络如图 9.36 所示。

这时新网络中边 $\langle a, b\rangle$ 的流 $f(a, b)=0$，如果作为路径的反向边，其长度 $w(b, a)=\infty$，故此时从 s 到 t 只有一条通路，即

$$P_3\ \ s\longrightarrow b\longrightarrow c\longrightarrow t$$

路径上的边均为前向边，可增加的流量为 1，并使 $\langle s, b\rangle$ 成为饱和边，得到如图 9.37 所示的新网络。

在新网络中，已不存在从 s 到 t 的通路，计算结束。

将图 9.34 中各边的容量减去图 9.37 中对应边的容量即为流经该边的流量，于是得到图 9.38。

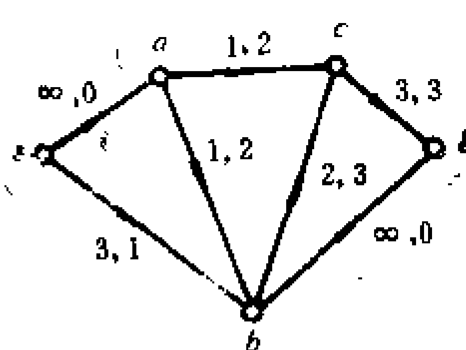


图 9.36

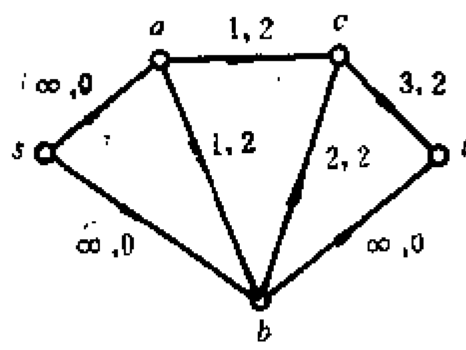


图 9.37

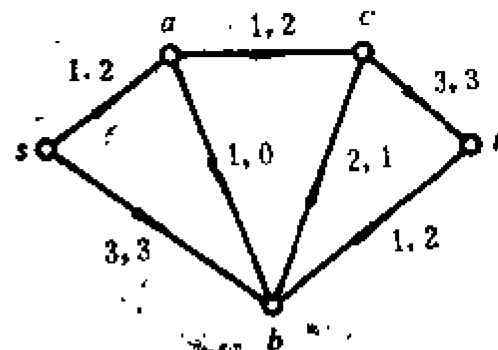


图 9.38

由图可得网上的最大流和费用

$$\max f(s, t)=2+3=5$$

$$\begin{aligned}a(G, F)&=1\times 2+3\times 3+1\times 2+2\times 1+3\times 3+1\times 2\\&=26\end{aligned}$$

这个费用就是输送流量为 5 时需要的最小费用。

二、对偶变量法

与最大流问题一样，最小费用流问题也是一个线性规划问题，因此可以应用线性规划

的对偶理论及互补松弛条件，在各结点 u 上设置一个对偶变量 $\pi(u)$（称为 u 的结点数），作为费用控制函数，使网络的增流过程严格按照最小费用增值的方向增加流量，直到达到最大流或预定的数值为止，可以证明此时输送流的费用也是最小的。限于篇幅，这里不准备介绍线性规划的有关内容，只不加证明地引入最小费用流算法如下：

1. for 所有的 $u\in V$ do $\pi(u)\leftarrow 0$
2. for 所有的 $\langle u, v\rangle\in E$ do $f(u, v)\leftarrow 0$
3. TEST⟶true
4. 用标记法找可增广路
5. if 收点 t 获得标记 then
   begin
6. 修改边的流量，若 $I=I'$ 计算结束
7. TEST←true
8. goto 4
   end
9. if 收点 t 得不到标记 then
   begin
10. if TEST then 若 I 使 N 饱和，计算结束
11. 修改结点数 $\pi(u)\leftarrow\pi(u)+1$
12. TEST←false
13. goto 4
   end

算法说明：

1. 算法的第 1 和第 2 行为初始化，这时网络上的流为零流，每一结点 u 的值 $\pi(u)$ 也都置零。

2. 第 4 行是用标记法寻找增广路，与前面讲的标记法类似，也是反复通过标记和增广过程不断增加网上的可行流，最后达到预定的数值或最大值，不过这里的标记过程是受到结点的 $\pi(u)$ 控制的，必须遵循如下规则：

开始时，除发点 S 外，其余所有结点都是未标记的结点。若 u 为已标记的结点，则与 u 邻接的结点 v 必须满足以下条件方可获得标记：

$$\left.\begin{array}{l} f(u,v)<c(u,v) \text{ 并且 } \pi(v)-\pi(u)=a(u,v) \\ \text{或者} \\ f(u,v)>0 \text{ 并且 } \pi(v)-\pi(u)=a(u,v) \end{array}\right\} \qquad (9.10)$$

3. 当收点 t 获得标记时，即找到了一条可增广路，可按以前的方法增加路上各边的流量，即算法中的第 6 行。

4. 若 t 不可能获得标记，则有两种可能情况，一是网络流已达到饱和，这时运算结束。另一种情况是还未达到饱和，这时应修改各结点的 $\pi(u)$ 值。回到第 4 行重复标记过程。

5. 在修改结点的 $\pi(u)$ 值时，为了简化计算，已标记的结点的 $\pi(u)$ 值不变，未标记的结点的 $\pi(u)$ 值等量递增（每次增加 1）

6. 算法第 6 行中的 I' 为预先给定的流 $f(s, t)$，如果事先没有给出这一数值，则运算在求得最大流时结束。

7. 算法中设置了一布尔变量 $TEST$，这是为了避免每当收点 t 不能获得标记时都要去检测网络是否达到饱和，有了这一布尔变量，则只有当上一次标记过程找到了可增广路，而这一次收点未能标记时，才需要去检查网络是否达到饱和，这样就可以节省计算的时间。

例 9.9 试用对偶变量法求图 9.34 所示的网络的最小费用流。

解：1. 初始状态各边的流均为零，且置

$$\pi(s)=\pi(a)=\pi(b)=\pi(c)=\pi(t)=0$$

s 已标记，与 s 邻接的点 a, b 均不满足式(9.10)的条件，因此不能标记，于是收点 t 也不能标记，此时显然网络未达到饱和，执行算法第 11 行，有

$$\pi(a)=\pi(b)=\pi(c)=\pi(t)=0+1=1$$

回到算法第 4 行，重新标记

2. 与 s 邻接的两个结点 a 和 b，由于

$$\pi(b)-\pi(s)=1\neq a(s, b)=3$$

$$\pi(a)-\pi(s)=1=a(s, a)\text{且 } f(s, a)<c(s, a)$$

所以 b 不能标记，a 可以标记，于是已标记的结点增加为 $\{s, a\}$ 但此时未标记的结点与已标记的结点之间均不满足式(9.10) 故 t 不能标记，且 $TEST=false$，不须检查网络是否达到饱和即可修改未标记的结点的值如下

$$\pi(b)=\pi(c)=\pi(t)=1+1=2$$

回到算法的第 4 行，重新标记

3. 此时结点 b, c 均满足式(9.10)，已标记结点增加为 $\{s, a, b, c\}$，t 仍不能标记，故

$$\pi(t)=2+1=3$$

回到算法第 4 行，重新标记

4. 未标记点 t 对已标记点 b 有

$$\pi(t)-\pi(b)=3-2=1=a(b, t)\text{且 } f(b, t)<c(b, t)$$

所以收点 t 获得标记，于是得到可增广路

$$s\longrightarrow a\longrightarrow b\longrightarrow t$$

增加这条路上的流到最大可能值 2，回到算法第 4 行，重新标记。

5. 此时 s 为已标记点，其余均为未标记点，用式(9.10)的条件检查，与 s 邻接的两个结点 a, b 均不能获得标记，因而收点也不可能获得标记，于是修改未标记结点的 $\pi(u)$ 值：

$$\pi(a)=1+1=2, \qquad \pi(b)=2+1=3$$

$$\pi(c)=2+1=3, \qquad \pi(t)=3+1=4$$

回到算法第 4 行，重新标记

6. 用式(9.10)的条件检查可得：由 s 可标记点 b，由 b 可标记点 a，由 a 可标记点 c，但 t 不能获得标记，于是对结点 t 修改：

$$\pi(t)=4+1=5$$

回到算法第 4 行，用式(9.10)检查，收点 t 仍然不能获得标记，于是再修改其值

$$\pi(t)=5+1=6$$

回到算法第 4 行，重新标记

7．用式(9.10)检查，可由 c 标记 t，于是得到一条可增广路为

$$s\longrightarrow b\longrightarrow a\longrightarrow c\longrightarrow t$$

其中 $\langle a, b\rangle$ 为反向边，增广路中流的增量为 2，回到算法第 4 行，重新标记。

8．可由 s 标记 b，但其余点不能标记，修改其 $\pi(u)$值，得到

$$\pi(a)=2+1=3,\ \pi(c)=3+1=4,\ \pi(t)=6+1=7$$

回到算法第 4 行，重新标记。

9．用式(9.10)检查，结点 a, c, t 仍然不能标记，再修改其值

$$\pi(a)=3+1=4,\ \pi(c)=4+1=5,\ \pi(t)=7+1\ =8$$

回到算法第 4 行，重新标记

10．用式(9.10)检查可得：由 b 可标记点 c，由 c 可标记点 t，于是得到一条可增广路

$$s\longrightarrow b\longrightarrow c\longrightarrow t$$

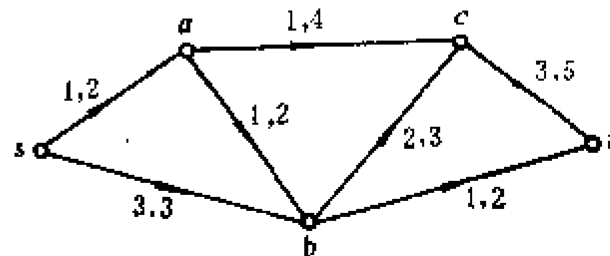
流的增量为 1。回到算法第 4 行，重新标记

11．收点 t 已不能标记，检查到网络已达到饱和，计算结束，得到最大流及最小费用为

$$\max f(s, t)=5,\ a(G, F)=26$$

上面的迭代过程可归纳成如表 9.1 所示，并将图 9.34 重新画在表上。

表 9.1



N：每条边 $\langle u, v\rangle$ 上标出 $a(u, v)$，$c(u, v)$

迭代	$\pi(s)$	$\pi(a)$	$\pi(b)$	$\pi(c)$	$\pi(t)$	标记通过的边	标记点	$f(s,a)$	$f(s,b)$	$f(s,c)$	$f(a,b)$	$f(b,c)$	$f(b,t)$	$f(c,t)$
0	0	0	0	0	0	无	s	0	0	0	0	0	0	0
1	0	1	1	1	1	(s,a)	s,a	[illegible]	0	0	0	0	0	0
2	0	1	2	2	2	$(s,a),(a,b),(a,c)$	s,a,b,c	0	0	0	0	0	0	0
3	0	1	2	2	3	$(s,a),(a,b),(a,c),(c,t)$	s,a,b,c,t	2	0	0	2	0	2	0
4	0	1	2	2	3	无	s	2	0	0	2	0	2	0
5	0	2	3	3	4	$(s,b),(b,a),(a,c)$	s,b,a,c	2	0	0	2	0	2	0
6	0	2	3	3	5	$(s,b),(b,a),(a,c)$	s,b,a,c	2	0	0	2	0	2	0
7	0	2	3	3	6	$(s,b),(b,a),(a,c),(c,t)$	s,b,a,c,t	2	2	2	0	0	2	2
8	0	2	3	3	6	(s,b)	s,b	2	2	2	0	0	2	2
9	0	3	3	4	7	(s,b)	s,b	2	2	2	0	0	2	2
10	0	4	3	5	8	$(s,b),(b,c),(c,t)$	s,b,c,t	2	3	2	0	1	2	3
11	0	4	3	5	8	无	s	网络最大流 = 5						

§9.7 有向图的中国邮路问题

上一章我们讨论了无向图的中国邮路问题，如果邮路需要用边带权的有向图来表示，就成为这一节我们要讨论的课题。与无向图比较，解有向图的中国邮路问题要复杂一些，但是下面我们将会看到，应用求最小费用流的算法来解这一问题，会取得很好的效果。

首先，如果有向图D是连通且平衡的，根据定理8.8，D是一个有向欧拉图，遍历欧拉回路，即是邮路的最优投递路线，它的算法已在§8.4中讲过。这里我们只讨论D不是有向欧拉图的一般情况。

与无向图类似，如果不是欧拉图，则邮路的投递路线，必须重复穿过一些边，对无向图来说，只要图是连通的，邮路问题一定可以通过重复一些边而得到解决。但是对于有向图，在某种情况下即使重复一些边，邮路问题也是无解的。如图9.39所示的有向图，不管哪些边重复多少遍，都不可能构成穿过每条边至少一次的有向闭合回路，从图可以看出原因在于不存在从结点子集$\{x_1, x_2, x_3\}$到结点子集$\{y_1, y_2, y_3\}$的通路。由此可知，一个有向图对邮路问题有解，必须满足某些条件，即下面的定理。

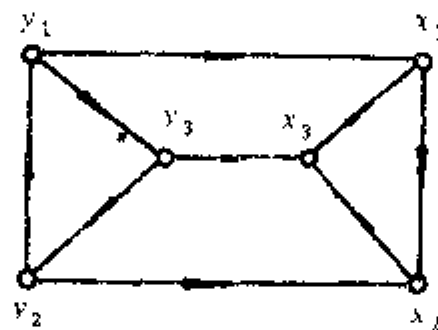


图 9.39

定理 9.4 有向图D存在穿过每条边至少一次的有向闭合回路的充要条件是D是强连通图。

证： 显然，如果D存在这样的回路，则D的任意两个结点u和v必然在回路上至少出现一次，因此一定存在从u到v的通路以及从v到u的通路，故D是强连通的。

反之，如果D是强连通图，则必然存在从D的任一结点u出发经过某些结点又回到u的闭合回路C，如果C已通过每条边至少一次，则命题得证，否则，设边$\langle v_i, v_j\rangle$不在C上，由于图是强连通的，必然存在从u到v_i的通路$P(u, v_i)$以及从v_j到u的通路$P(v_j, u)$，这两条通路加上边$\langle v_i, v_j\rangle$构成回路C'

$$C' = (P(u, v_i), \langle v_i, v_j\rangle, P(v_j, u))$$

将C与C'并，得到含有边$\langle v_i, v_j\rangle$的一个更大的回路C''，如果还有边不在C''上，又可继续上面的过程，最终即可得到通过每条边至少一次的闭合回路。

下面我们在满足强连通的前提下，讨论有向图的中国邮路问题。 ▮

如果强连通有向图不是欧拉图，邮递路线必须重复穿过某些边，设边$\langle u, v\rangle$重复穿过的次数记作$r(u, v)$，则最优邮递路线的问题就是寻求重复边的总长度为最小值，即

$$\min\Big(\sum_{\langle u, v\rangle} d(u,v)r(u,v)\Big) \qquad (A)$$

其中$d(u, v)$表示边$\langle u, v\rangle$的长度（权）。

重复穿过某些边，相当于在原来的图D上添加这些边的平行边，使图成为欧拉图D''，所以添加平行边，对无向图来说目的在于使每个结点的次数均为偶数，对有向图来说，则是使之成为平衡图（即每一结点的引入次数与引出次数相等），为此我们用$D(u)$表示结点u的引入次数与引出次数之差，即

$$D(u) = deg^-(e) - deg^+(u)$$

因此，当$D(u)>0$时，在构造欧拉图D''中必然在结点u上添加引出边以增加u的引出次数$deg^+(u)$，同理当$D(u)<0$时，则在u上添加引入边以增加u的引入次数使之达到平衡。如果我们把边的重复次数$r(u,v)$看作边的流量，$d(u,v)$看作边$\langle u,v\rangle$通过单位流量的费用，则式（A）寻求重复边总长度为最小的问题可化归为求最小费用流的问题。

对任一有向图，下式均成立

$$\sum_{u,D(u)<0} D(u) = -\sum_{v,D(v)>0} D(v)$$

因此我们可以构造一个有向网络D'，令$D(u)>0$的结点u为发点，$D(v)<0$的结点v为收点，一般来说这些结点不是唯一的，可以仿照多发点多收点的算法，增设一个虚拟发点S和一个虚拟收点t，连接S和各发点u，置$<S, u>$的容量为$D(u)$，费用为0，同理，连接各收点v和t，置$\langle v,t\rangle$的容量为$-D(v)$，费用亦为0，其他各边的容量均为∞，费用即为该边的长度（权）。于是可求出D'的最小费用流，它构成了总长度为最小的重复边，因而可求出有向欧拉图D''，找到了邮递路线的最优解。

根据上面的分析，即得出有向图中国邮路问题的算法如下：

1．构造有向网络D'

2．求出D'的最小费用的最大流

3．构造有向欧拉图D''

4．求出D''的有向欧拉回路，即是邮递路线的最优解。

例 9.10 设有向邮路如图 9.40 所示，求最优邮递路线。（图中边的数字表示长度）

解：

（一）构造D′

$$D(v_1)=deg^-(v_1)-deg^+(v_1)$$
$$=1-2=-1$$
$$D(v_2)=deg^-(v_2)-deg^+(v_2)=2-2=0$$

同理　$D(v_3)=2$，$D(v_4)=1$，$D(v_5)=-2$

因此有两个发点：v_3，v_4，两个收点：v_1，v_5。为此增加一虚拟发点S和一虚拟收点t，连接S和v_3，v_4，及v_1，v_5和t得D'如图 9.41，图中边上的数字表示单位流量的费用，并置

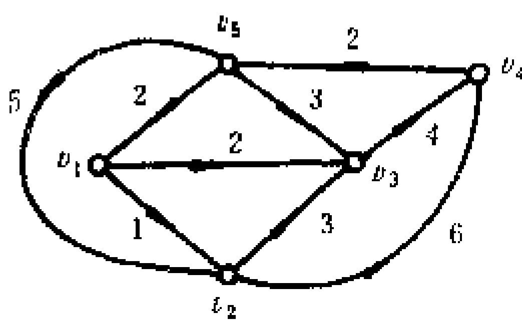


图 9.40

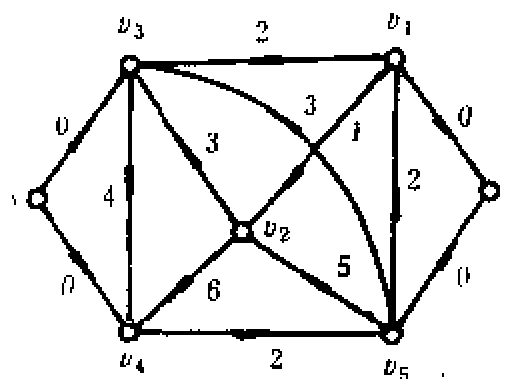


图 9.41

$$C(S,v_3)=C(v_5,t)=2$$
$$C(S,v_4)=C(v_1,t)=1$$

对所有其他边$\langle u,v\rangle$，均置$C(u,v)=\infty$

（二）求 D′ 中最小费用最大流

用§9.6中的算法，当饱和时得到网上流的两条路径

P_1：$S \to v_3 \to v_4 \to v_5 \to t$，流量为 2

P_2：$S \to v_4 \to v_5 \to v_1 \to t$，流量为 1

（三）构造欧拉图 D″

将路径 P_1, P_2 的边加到 D 上（这里 S，t 是虚拟结点，故路径两端的边需除去），注意：P_1 的流量为 2，故这些边需重复两次，而 P_2 上的边只需重复 1 次，于是得到 D'' 如图 9.42所示。

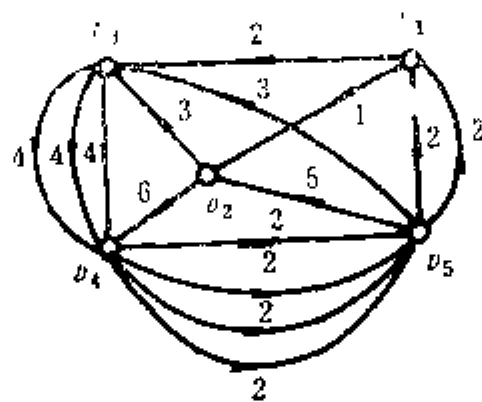


图 9.42

（四）最优邮递路线为

$$(v_1, v_2, v_3, v_4, v_5, v_2, v_4, v_5, v_3, v_4, v_5, v_1, v_3, v_4, v_5, v_1)$$

由于算法中各步的时间复杂性都是 n 或 $|E|$ 的多项式，总的计算量也是多项式级的，因此是一较好的算法。

§9.8 无向网络的流

一、流与切割

无向边权网络与有向边权网络的区别，主要在于前者的每一条边（u，v）对通过流的方向是不加限制的，既可从 u 流向 v，也可从 v 流向 u，因此我们在图中给边标方向只是表示目前边中流的方向。与有向边权网络相同，我们仍然讨论受容量限制的可行流。

在第三章§3.9中，我们讲了断集的概念，在这里我们要讨论一种特殊的断集，即在无向网络 $N=(V, E)$ 中，若发点 $S \in V_1 \subset V$，收点 $t \in \bar{V}_1$，则称断集 $E(V_1, \bar{V}_1)$ 为 N 的切割，并把由 V_1 到 $\bar{V}_1$ 的方向定为此切割的方向。

定义 9.14 设 F 是无向网络 $N=(V, E)$ 上分配的可行流，若存在切割 $E(V_1, \bar{V}_1)$，它的每一条边都是饱和边，则称此切割为当前流的饱和切割。

定义 9.15 设 F 是无向网络 $N=(V, E)$ 上分配的可行流，若切割 $E(V_1, \bar{V}_1)$ 的每一条边都是饱和边，且边中流的方向均和切割的方向一致，则称此切割为当前流的基本饱和切割。

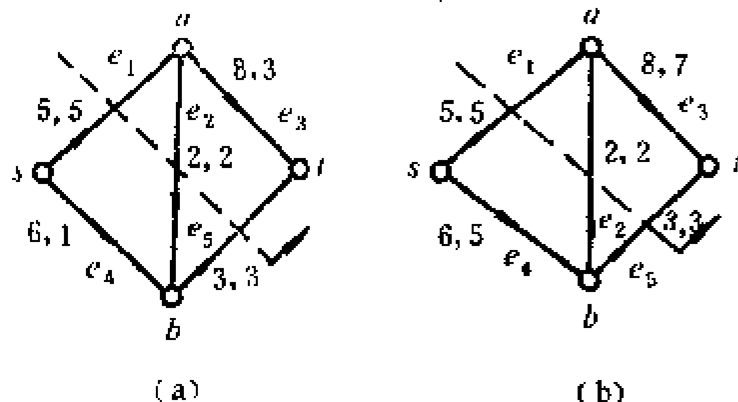


图 9.43

必须指出，饱和切割与基本饱和切割是有区别的，如图9.43（a），切割 $S=\{e_1, e_2, e_5\}$ 是饱和切割但不是基本饱和切割，因为边 e_2 中流的方向与切割的方向不一致，而图 9.43（b）的切割 $\{e_1, e_2, e_5\}$ 则是基本饱和切割。

定理 9.5 设 F 为网络 $N=(V, E)$ 上的流，若不存在基本饱和切割，则流量必小于网的最大流，即

$$f(s, t) < f_m$$

其中 f_m 表示网络可能传输流的最大值。

证 对网上流F的分配，分两种情况证明如下

（1）既不存在基本饱和切割，也不存在饱和切割。

此时，我们将各条边（u，v）的容量C（u，v）改为

$$C'(u,v)=C(u,v)-f(u,v)$$

并删除$C'(u,v)=0$的那些边，由于不存在饱和切割，图仍然是连通的，即存在从S到t的通路P_{St}，我们可以在这条通路上增加一流量Δf

$$\Delta f=\min\{C'(u,v)\mid(u,v)\in P_{St}\}$$

由此可见原来的流量f（s，t）并非最大流。

（2）不存在基本饱和切割，但存在饱和切割。

如图9.44所示的$S=E(V_1,\bar{V}_1)$表示饱和切割，其中$e_1,e_2\cdots,e_i$表示流的方向与切割方向一致的饱和边，$e_{i+1},e_{i+2},\cdots e_K$表示流的方向与切割方向相反的边，记$S_2=\{e_{i+1},\ e_{i+2}\cdots e_K\}$

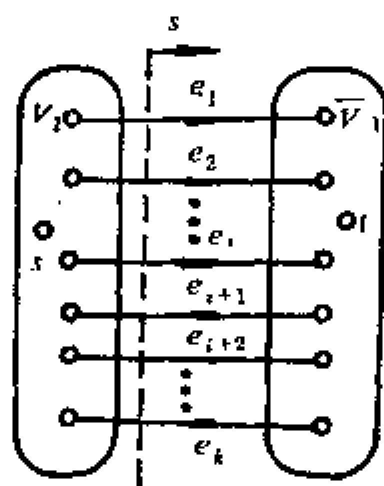


图 9.44

现在我们将各条边的容量改为：

$$C'(u,v)=C(u,v)-f(u,v),\ \text{若}(u,v)\notin S_2$$

$$C'(u,v)=C(u,v)+f(u,v),\ \text{若}(u,v)\in S_2$$

然后删除$C'(u,v)=0$的边，显然图仍然是连通的，即存在从S到t的通路P，可以增加这条通路中的流，因此原来的流量$f(S,t)$不是最大流。 ▌

定理 9.6 当且仅当网上的流F至少产生一个基本饱和切割时，才有

$$f(s,t)=f_m$$

证 对网络N的任一切割E（$V_1,\bar{V}_1$），网上的流都必须通过切割中的边，因此从S到t能输送的最大流量不能超过切割中边的容量之和，这对任一切割均不例外，而基本饱和切割各边的流量已达到正向饱和，显然此时的流量已达到最大可能值。

由定理9.5可知，只要未出现基本饱和切割，网上的流还可增加，作为定理的逆命题，即证当网上的流为最大值时，一定产生了至少一个基本饱和切割。 ▌

由定理9.6即可得出无向边权网络的最大流最小切割定理如下

定理 9.7 无向边权网络的最大流等于切割的最小容量，即

$$\max f(s,t)=\min\{C(K)\mid K\in\{E(V_1,\bar{V}_1)\}\} \tag{9.11}$$

二、最小切割的求法

定理9.7告诉我们，只要找出网络的最小切割，即可确定网络可能输送的最大流量，诚然，我们可以把网络的所有断集都求出来，然后摒弃那些没有分离S和t的断集，结果得到所有分离S和t的切割，从中找出容量最小的切割，即为所求。但是这种做法效率是低的。下面我们介绍应用生成元产生所有切割的算法，为此先介绍两种特殊断集环和的概念。

对于无向边权网络$N=(V,E)$，$V_1\subset V$，$\bar{V}_1=V-V_1$，$V_1\neq\phi$，若发点$S\in V_1$，收点$t\in\bar{V}_1$，则将切割记作$S(s;t)$。若$s,t\in V_1$（或$s,t\in\bar{V}_1$）则将断集记作$S(st;\cdot)$。因此前者表示分离S

和 t 的断集，后者则表示没有分离 S 和 t 的断集。以后我们将证明，一个断集如果不是割集，则可分解成为若干个割集的边不重并集，现在我们先讨论断集是割集的情况。

定理 9.8 设 $S_1(s;t)$和 $S_2(s;t)$都是割集，则

（1）若 $S_1(s;t)\oplus S_2(s;t)$是割集，则它不能分离 S 和 t。

（2）若 $S_1(s;t)\oplus S_2(s;t)$不是割集，则它可表为若干个边不重的割集的并集，而且或者有两个割集分离 S 和 t，或者一个也没有。

证：（1）我们将图的结点集合 V 分成四个互不相交的非空子集V_{11}，V_{12}，V_{21}，V_{22}，如图9.45所示，并令

$$V_1=V_{11}\cup V_{12}$$
$$V_2=V_{11}\cup V_{21}$$

则 $\bar{V}_1=V_{21}\cup V_{22}$，$\bar{V}_2=V_{12}\cup V_{22}$

令 $s\in V_{11}$，$t\in V_{22}$

则 $S_1(s;t)=E(V_1\times\bar{V}_1)$，$S_2(s;t)=E(V_2\times\bar{V}_2)$

图 9.45

由定理 3.20 的推论可知

$$E(V\times V)=E(V_1\times V_1)\oplus E(V_1\times\bar{V}_1)\oplus E(\bar{V}_1\times\bar{V}_1)$$

故 $E(V_1\times\bar{V}_1)=E(V\times V)\oplus E(V_1\times V_1)\oplus E(\bar{V}_1\times\bar{V}_1)$

同理 $E(V_2\times\bar{V}_2)=E(V\times V)\oplus E(V_2\times V_2)\oplus E(\bar{V}_2\times\bar{V}_2)$

则 $S_1(s;t)\oplus S_2(s;t)=E(V_1\times\bar{V}_1)\oplus E(V_2\times\bar{V}_2)$

$$=E(V_1\times V_1)\oplus E(\bar{V}_1\times\bar{V}_1)\oplus E(V_2\times V_2)\oplus E(\bar{V}_2\times\bar{V}_2)$$

以 $V_1=V_{11}\cup V_{12}$，$\bar{V}_1=V_{21}\cup V_{22}$，$V_2=V_{11}\cup V_{21}$，$\bar{V}_2=V_{12}\cup V_{22}$ 代入

$$S_1(s;t)\oplus S_2(s;t)=E(V_{11}\cup V_{12}\times V_{11}\cup V_{12})\oplus E(V_{21}\cup V_{22}\times V_{21}\cup V_{22})$$
$$\oplus E(V_{11}\cup V_{21}\times V_{11}\cup V_{21})\oplus E(V_{12}\cup V_{22}\times V_{12}\cup V_{22})$$

因 $E(V_{11}\cup V_{12}\times V_{11}\cup V_{12})=E(V_{11}\times V_{11})\oplus E(V_{11}\times V_{12})\oplus E(V_{12}\times V_{11})\oplus E(V_{12}\times V_{12})$

类似地，其他各项亦可展开，经过化简后可得

$$S_1(s;t)\oplus S_2(s;t)=E(V_{11}\times V_{12})\oplus E(V_{21}\times V_{22})\oplus E(V_{11}\times V_{21})\oplus E(V_{12}\times E_{22})$$
$$=E(V_{11}\cup V_{22}\times V_{12}\cup V_{21})$$
$$=E(V_3\times\bar{V}_3)$$

其中 $V_3=V_{11}\cup V_{22}$，$\bar{V}_3=V_{12}\cup V_{21}$，$V_3\cup\bar{V}_3=V$

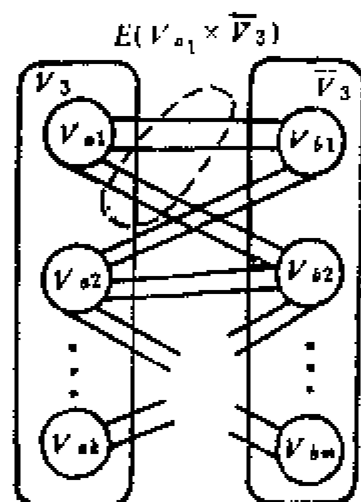


图 9.46

因 $E(V_3\times\bar{V}_3)$是割集，而$s,t\in V_3$，故它不分离 s 和 t。

（2）由于 $E(V_3\times\bar{V}_3)$ 不是割集，设删除断集 $E(V_3\times\bar{V}_3)$ 的所有边后，V_3 被分成K个连通分块而$\bar{V}_3$ 被分成m个连通分块，如图9.46所示。这时有两种可能

（a）s，t 同在一个连通分块中，不失一般性，设

$$s,t\in V_{a1}$$

则

$$S_1(s;t)\oplus S_2(s;t)$$
$$=E(V_3\times\bar{V}_3)$$
$$=\bigoplus_{i=1}^{K}E(V_{ai}\times\bar{V}_3)$$
$$=E(V_{a1}\times\bar{V}_3)\oplus E(V_{a2}\times\bar{V}_3)\oplus\cdots\oplus E(V_{aK}\times\bar{V}_3)$$

$$=S_1(st;\cdot)\cup S_2(st;\cdot)\cup\cdots\cup S_K(st;\cdot) \qquad (A)$$

它们是边不重的割集的并集，而且没有一个割集分离了 s 和 t。

（b）s，t 处在两个不同的连通分块中，设

$$s\in V_{a1},\ t\in V_{a2} \text{ 且 } V_{a1}\cap V_{a2}=\phi$$

由式（A）可见，只有两个割集 $E(V_{a1}\times\bar{V}_3)$和 $E(V_{a2}\times\bar{V}_3)$分离 s 和 t，其余割集都没有分离 s 和 t。 ▌

定理 9.9 设 $S_1(s;t)$和 $S_2(st;\cdot)$都是割集，则

（1）若 $S_1(s;t)\oplus S_2(st;\cdot)$是割集，则它分离 s 和 t。

（2）若 $S_1(s;t)\oplus S_2(st;\cdot)$不是割集，则它可表为若干边不重的割集的并集，而且其中必有一个割集分离 s 和 t。

证：（1）仍将图的结点集合V分为四个互不相交的非空子集，如图9.45所示，并令

$$S\in V_{11},\ t\in V_{21},$$

$$E(V_1\times\bar{V}_1)=S_1(s;t),\ E(V_2\times\bar{V}_2)=S_2(st;\cdot)$$

因 $$S_1(s;t)\oplus S_2(st;\cdot)=E(V_3\times\bar{V}_3)$$

可见 $$s\in V_3,\ t\in\bar{V}_3$$

即 $E(V_3\times\bar{V}_3)$分离了 s 和 t。

（2）由于 $E(V_3\times\bar{V}_3)$不是割集，图的结构仍用图 9.46 来表示，不失一般性，设

$$S\in V_{a1},\ t\in V_{b1}$$

因 $$E(V_3\times\bar{V}_3)=E(V_{a1}\times\bar{V}_3)\oplus E(V_{a2}\times\bar{V}_3)\oplus\cdots\oplus E(V_{aK}\times\bar{V}_3)$$

而 $$V_{ai}\cap V_{aj}=\phi \quad i\neq j,\ i,j=1,2\cdots K$$

故 $$E(V_3\times\bar{V}_3)=E(V_{a1}\times\bar{V}_3)\cup E(V_{a2}\times\bar{V}_3)\cup\cdots\cup E(V_{aK}\times\bar{V}_3)$$

即 $E(V_3\times\bar{V}_3)$可表为若干边不重的割集的并集，而且其中只有割集 $E(V_{a1}\times\bar{V}_3)$分离了 s 和 t。 ▌

定理 9.10 设 $S_1(st;\cdot)$和 $S_2(st;\cdot)$都是割集，则

（1）若 $S_1(st;\cdot)\oplus S_2(st;\cdot)$是割集，则它不分离 s 和 t。

（2）若 $S_1(st;\cdot)\oplus S_2(st;\cdot)$不是割集，则它可表为若干边不重的割集的并集，而且或者有两个割集分离 s 和 t，或者一个也没有。

证：（1）我们将结点集合V分成三个互不相交的非空子集V_1，V_2，和V_3，如图9.47所示。并设

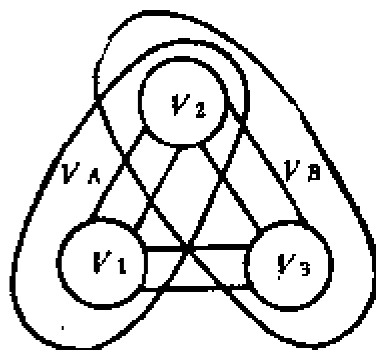


图 9.47

$$V_A=V_1\cup V_2,\ V_B=V_2\cup V_3$$

$$s,t\in V_2$$

则 $E(V_A\times\bar{V}_A)$和 $E(V_B\times\bar{V}_B)$都是没有分离 s 和 t 的割集，即

$$S_1(st;\cdot)=E(V_A\times\bar{V}_A)$$

$$S_2(st;\cdot)=E(V_B\times\bar{V}_B)$$

则 $$S_1(st;\cdot)\oplus S_2(st;\cdot)=E(V_A\times\bar{V}_A)\oplus E(V_B\times\bar{V}_B)$$

$$=E(V_1\cup V_2\times V_3)\oplus E(V_2\cup V_3\times V_1)$$

$$=E(V_1\times V_3)\oplus E(V_2\times V_3)\oplus E(V_2\times V_1)\oplus E(V_3\times V_1)$$

$$=E(V_2\times V_3)\oplus E(V_2\times V_1)$$

$E(V_2 \times V_1 \cup V_3)$

由图可见，因 S，$t \in V_2$，故 $E(V_2 \times V_1 \cup V_3)$ 没有分离 S 和 t。 ▮

（2）若 $E(V_2 \times V_1 \cup V_3)$ 不是割集，仍可用图 9.46来表示，此时 V_2 相当于图中的 V_3，$V_1 \cup V_3$ 相当于图中的 $\bar{V}_3$，证明方法与定理9.8（2）的证明完全相同。

上面的定理同时也证明了一个断集或者是割集，或者是若干个边不重的割集的并集。因此我们可以把以上三个定理综合成如下定理。

定理 9.11 设 $\{S(s;t)\}$ 表示所有分离 S 和 t 的切割集合，$\{S(st;\cdot)\}$ 表示所有不属于 $\{S(s;t)\}$ 的断集集合，则

（1）若 S_1，$S_2 \in \{S(s;t)\}$，则 $S_1 \oplus S_2 \in \{S(st;\cdot)\}$

（2）若 S_1，$S_2 \in \{S(st;\cdot)\}$，则 $S_1 \oplus S_2 \in \{S(st;\cdot)\}$

（3）若 $S_1 \in \{S(s;t)\}$，$S_2 \in \{S(st;\cdot)\}$，则 $S_1 \oplus S_2 \in \{S(s;t)\}$。

根据定理9.11，如果我们取网络各结点的关联集 S_0，S_1，…，S_{n-2} 作为一组生成元，设

$$S_0 \in \{S(s;t)\}$$

$$S_i \notin \{S(s;t)\},\quad i=1,2,\cdots,n-2$$

则有 $\quad S_0 \oplus S_i \in \{S(s;t)\}$

即通过这组生成元的所有可能的环和，可以生成 $\{S(s;t)\}$。用 $C(S)$ 表示切割 S 的容量，可以求出最小切割，它的容量即为网络允许输送的最大流。于是我们得到求网络最大流的算法如下

1．求出图的线性独立关联集合组 $S_0, S_1, \cdots, S_{n-2}$。其中 $S_0 \in \{s;t)\}$，$S_i \notin \{S(s;t)\}$，$i=1,\cdots n-2$。

2．求出 $\{S(s;t)\}=\{S_0 \oplus S_K\}$。其中 S_K 表示集合 $\{\phi, S_1, S_2, \cdots, S_{n-2}\}$ 所有子集的环和

3．求出 $\max f(s,t)=\mathrm{mim}\{C(S) \mid S \in \{S(s;t)\}\}$

例 9.11 一无向边权网络如图 9.48 所示，其中边上的数字表示其容量，求网络的最大流。

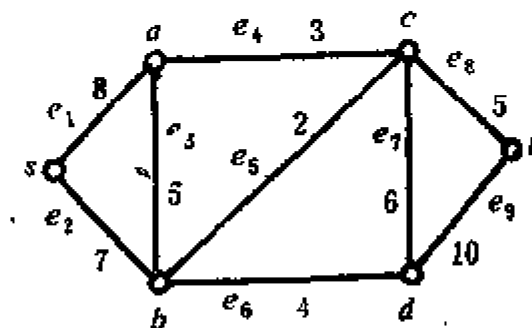


图 9.48

解：求出各点关联集

$$S(s)=\{e_1,\ e_2\}$$

$$S(a)=\{e_1,\ e_3,\ e_4\}$$

$$S(b)=\{e_2,\ e_3,\ e_5,\ e_6\}$$

$$S(c)=\{e_4,\ e_5,\ e_7,\ e_8\}$$

$$S(d)=\{e_6,\ e_7,\ e_9\}$$

则 $\quad S_0=S(s)\oplus\phi=S(s)$，$C(S_0)=7+8=15$

$S_1=S(s)\oplus S(a)=\{e_2,\ e_3,\ e_4\}$，$C(S_1)=15$

$S_2=S(s)\oplus S(b)=\{e_1,\ e_3,\ e_5,\ e_6\}$，$C(S_2)=19$

$S_3=S(s)\oplus S(c)=\{e_1,\ e_2,\ e_4,\ e_5,\ e_7,\ e_8\}$，$C(S_3)=31$

$S_4=S(s)\oplus S(d)=\{e_1,\ e_2,\ e_6,\ e_7,\ e_9\}$，$C(S_4)=35$

$S_5=S(s)\oplus S(a)\oplus S(b)=\{e_4,\ e_5,\ e_6\}$，$C(S_5)=9$

$S_6=S(s)\oplus S(a)\oplus S(c)=\{e_2,\ e_3,\ e_5,\ e_7,\ e_8\}$，$C(S_6)=25$

$$S_7=S(s)\oplus S(a)\oplus S(d)=\{e_2,\ e_3,\ e_4,\ e_6,\ e_7,\ e_9\},\quad C(S_7)=35$$

$$S_8=S(s)\oplus S(b)\oplus S(c)=\{e_1,\ e_3,\ e_4,\ e_6,\ e_7,\ e_8\},\quad C(S_8)=3$$

$$S_9=S(s)\oplus S(b)\oplus S(d)=\{e_1,\ e_3,\ e_5,\ e_7,\ e_9\},\quad C(S_9)=31$$

$$S_{10}=S(s)\oplus S(c)\oplus S(d)=\{e_1,\ e_2,\ e_4,\ e_5,\ e_6,\ e_8,\ e_9\},\quad C(S_{10})=39$$

$$S_{11}=S(s)\oplus S(a)\oplus S(b)\oplus S(c)=\{e_6,\ e_7,\ e_8\},\quad C(S_{11})=15$$

$$S_{12}=S(s)\oplus S(a)\oplus S(b)\oplus S(d)=\{e_4,\ e_5,\ e_7,\ e_9\},\quad C(S_{12})=21$$

$$S_{13}=S(s)\oplus S(b)\oplus S(c)\oplus S(d)=\{e_1,\ e_3,\ e_4,\ e_8,\ e_9\},\quad C(S_{13})=31$$

$$S_{14}=S(s)\oplus S(a)\oplus S(c)\oplus S(d)=\{e_2,\ e_3,\ e_5,\ e_6,\ e_8,\ e_9\},\quad C(S_{14})=36$$

$$S_{15}=S(s)\oplus S(a)\oplus S(b)\oplus S(c)\oplus S(d)=\{e_8,\ e_9\},\quad C(S_{15})=15$$

故 $\min S=S_5=\{e_4,\ e_5,\ e_6\}$

$$\max f(s,\ t)=C(S_5)=9$$

用计算最小切割的方法求网络的最大流，当网络的结点数为n时，有$n-1$个线性独立的关联集，需要进行 2^{n-2} 次环和运算，显然计算的复杂性不是多项式的，只有当n较小时，这一算法方才可行。

三、用标记法求最大流

标记法同样可以用来计算无向边权网络的最大流，只是对反向边在标记或增广过程中的算法要略作修改，事实上无向网络边中流的方向是没有限制的，所以反向边只是表示当前边中流的方向与路径方向相反。如图9.49所示，若流的方向定为由a流向c，则$f(a,$

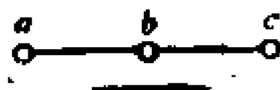


图 9.49

$b)$为正，边$(a,\ b)$称前向边，$f(b,\ c)$为负值，边$(b,\ c)$称反向边，于是标记过程中A_2的（2）作如下修改

（2）′ 若 $(u,\ v)\in E$ 且$f(u,\ v)<0$时，则将v标记为$(u,\ -,\ \Delta(v))$，其中

$$\Delta(v)=\min\{\Delta(u),\ C(u,\ v)-f(u,\ v)\}$$

对增广过程中 B_2 亦作部分修改如下：

若z的标记为$(q,\ -,\ \Delta(z))$，则

$$f(q,\ z)\longleftarrow f(q,\ z)+\Delta(t)$$

例 9.12 用标记法求图 9.48 所示的无向边权网络的最大流。

解：给出一初始可行流的分配如图9.50（a）所示，图中边上标的方向表示当前边中流的方向，标记及增广过程依次如图（a），（b），（c），（d）所示。在图（d）中，结点c，d和收点t已不可能获得标记，算法至此结束。可得

$$V_1=\{s,\ a,\ b\},\ \bar{V}_1=\{c,\ d,\ t\}$$

最小切割

$$(V_1,\ \bar{V}_1)=\{(a,\ c),\ (b,\ c),\ (b,\ d)\}$$

最大流

$$\max f(s,\ t)=C(V_1,\ \bar{V}_1)$$
$$=3+2+4=9$$

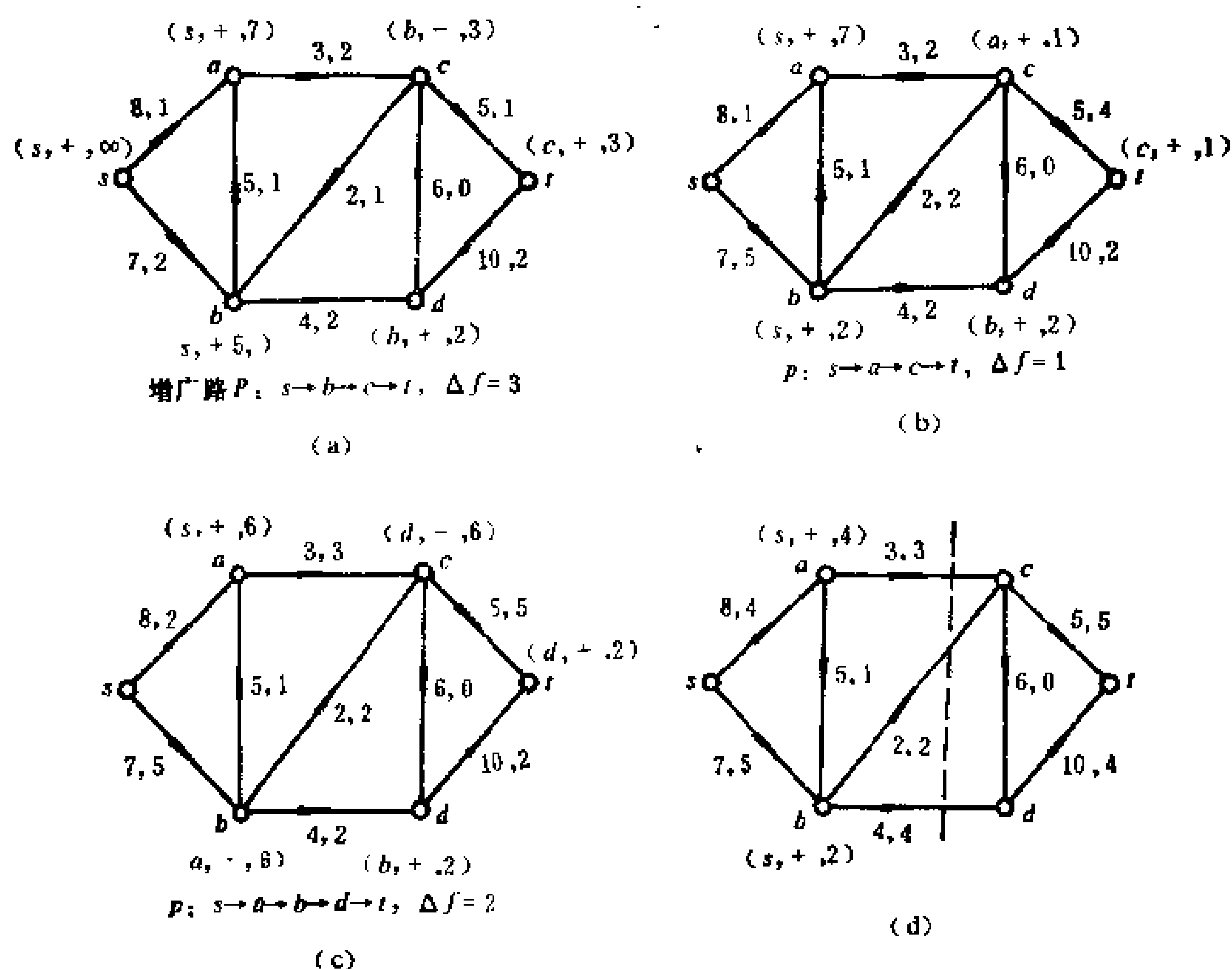


图 9.50

习题与思考题

1．在什么条件下，从发点至收点，能有一条全部由反向边组成的流的可增广路？在什么条件下，从发点至收点，能有一条全部由前向边组成的流的可增广路？

2．用标记法求图 9.51 所示网络的最大流（图中边的标号为容量及流量）

3．用最短路径法求图 9.52 网络的最大流

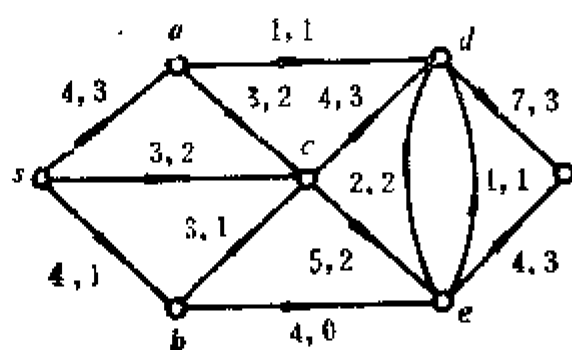


图 9.51

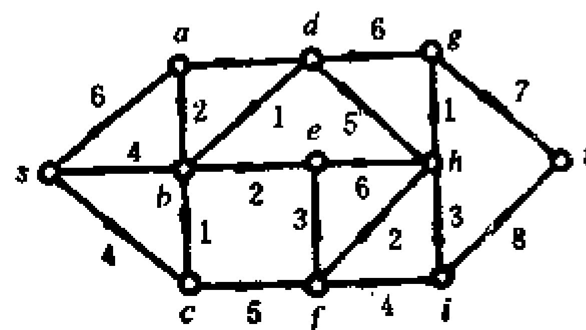


图 9.52

4．一制造商需要把两个车间 D_1，D_2 生产的同类商品通过运输网络输送到三个销售点 M_1，M_2，M_3 去，如图 9.53 所示，设各销售点计划销售量 $\phi(M)$ 分别为 $\phi(M_1)=10$，$\phi(M_2)=8$，$\phi(M_3)=8$，问如图所示网络的运输能力能否满足这一要求？两个车间生产数量多少最为恰当？

5. 七种设备要用五架飞机运往目的地，每种设备各有四台，这五架飞机的容量分别是 8，8，5，4，4 台，问能否有一种装载法，使同一种类型设备不会有两台在同一架飞机上。

6. 上题中若飞机的容量分别是 7，7，6，4，4 台，求问题的解。

7. 设 x_1，x_2，x_3 是三家工厂，y_1，y_2，y_3 是三个仓库，工厂生产的同类产品要运往仓库，其运输网络如图 9.54 所示，设 x_1，x_2，x_3 的生产能力分别为 40，20，10个单位，问应如何安排生产？

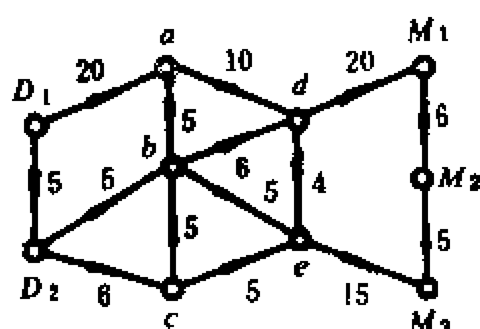


图 9.53

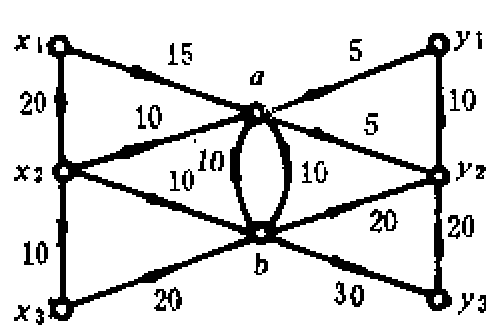


图 9.54

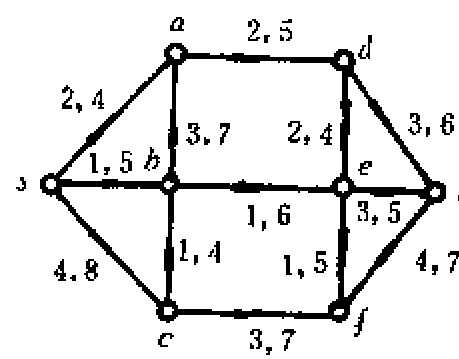


图 9.55

8. 设有张三、李四、王二、赵五四人及小提琴、大提琴、钢琴和吉他四种乐器，已知四人的擅长如下：

张三擅长拉小提琴、大提琴和吉他

李四擅长拉小提琴和大提琴

王二擅长拉大提琴和钢琴

赵五只会弹吉他

今假设四人同台演出，每人奏一种乐器，问四人同时各演奏一种乐器时所有可能的方案，试把此问题化为最大流问题。

9. 试用最短路径法求图 9.55 所示网络的最小费用流。（图中边上标的数字表示 $a(u, v)$和$c(u,v)$）

10. 试用对偶变量法解第 9 题。

11. 如图 9.56 的强连通有向图，试求最优邮递路线及其总路程。

12. 试用求最小切割的方法求图9.57所示无向边权网络的最大流。

13. 试用标记法求图 9.58 所示无向边权网络的最大流。

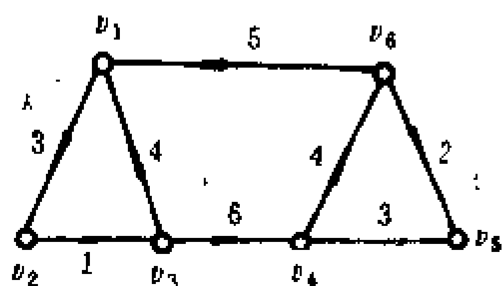


图 9.56

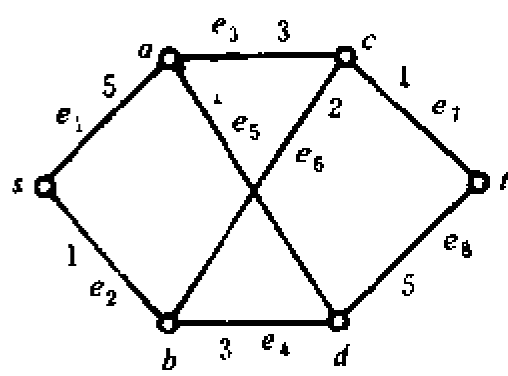


图 9.57

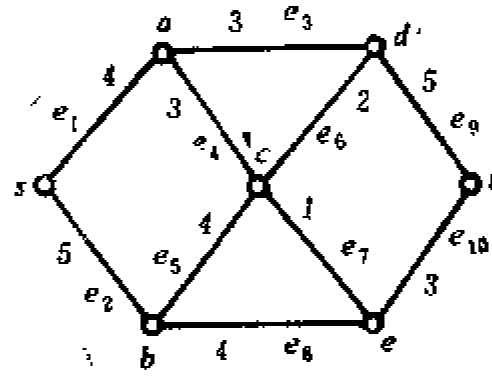


图 9.58

符 号 表

$deg(v)$	结点 v 的次数（与 v 关联的边数）
$deg^+(v)$	结点 v 的引出次数（引出弧的数目）
$deg^-(v)$	结点 v 的引入次数（引入弧的数目）
K_n	n 个结点的完全（无向）图
$\bar{G}$	图 G 的补图
$G_1 \oplus G_2$	图 G_1 和 G_2 的环和
$G_1 \cong G_2$	图 G_1 与 G_2 同构
$d(u,v)$	结点 u 到 v 的距离（最短路径长度）
$W(T)$	带权树 T 的权（树叶带权）
$\lfloor x \rfloor$	小于等于 x 的最大整数
$\lceil x \rceil$	大于等于 x 的最小整数
$S(v)$	结点 v 的关联集（与 v 关联的边集）
K_T	图 G 关于生成树 T 的基本割集组
$deg(R)$	面 R 的次数（R 周界边的数目）
$\delta(G)$ 或 δ	图 G 的最小次数，$\min\{degv \mid v \in V\}$
$\Delta(G)$ 或 Δ	图 G 的最大次数，$\max\{degv \mid v \in V\}$
$\theta(G)$	图 G 的厚度（非平面图分为平面子图的最少数目）
$K_{m,n}$	完全偶图，$m = \|V_1\|$，$n = \|V_2\|$
$E(P)$	路径 P 上的边集
$N(A)$	与 A 中结点邻接的点集
$A(v)$ 或 $Adj(v)$	与 v 邻接的结点集
$w(u,v)$	边 (u, v) 或 $\langle u, v \rangle$ 带的权
$w(M)$	匹配 M 边权之和
$\eta(G)$	图 G 的奇分支数
$\psi_v(G)$	图 G 的点色数
$\psi_e(G)$	图 G 的边色数
$I(G)$	G 的独立数（最大独立集的基数）
$D(G)$	G 的支配数（最小支配集基数的最小值）
$C(G)$	G 的点覆盖数（最小点覆盖的基数）
$B(G)$	G 的团的最小数
$J(G_1, G_2)$	G_1 和 G_2 的联图
$P_K(G)$	G 的不同 K 着色数（G 的着色多项式）
$TE(v_j)$	事件 v_j 的最早完成时间
$TL(v_j$	事件 v_j 的最晚完成时间

G^+	G的闭包图
$C(i,\ j)$	边$\langle v_i, v_j\rangle$的容量
$f(i,j)$	边$\langle v_i, v_j\rangle$上的流量
$f(s,t)$	网络中的流量
$C(V_1,\ \bar{V}_1)$	切割（V_1，$\bar{V}_1$）的容量
Δ_0	网络的可增广路中流的增量
Δ_i	边$\langle v_i,\ v_{i+1}\rangle$的可增加流量
$C(v_i)$	点权网络结点v_i的容量
$a(u,\ v)$	网络中流过边$\langle u, v\rangle$单位流量的费用

参 考 文 献

〔1〕 Alan Gibbons, Algorithmic graph theory, Cambridge University Press, 1985.

〔2〕 J. A. Bondy and U. S. R. Murty, Graph theory with Applications, The Macmillan press LTD, 1976.

〔3〕 Wai-Kai Chen, Applied graph theory, North-Holland Publishing Company, 1976.

〔4〕 Wataru Mayeda, Graph theory, Wiley-Interscience, 1972.

〔5〕 Edward Minieka, Optimization Algorithms for Network and Graph, Marcel Dekker, Inc, New York, 1978.

〔6〕 R. T. Rockafellar, Network flows and monotropic optimization, John Wiley & Sons, Inc, 1984.

〔7〕 Michael Capobiaco and John C. Molluzzo, Examples and Counterexamples in Graph theory, North-Holland. New York, 1978.

〔8〕 F. Harary, Graph Theory, Addisonwesley Publishing Co., Inc., 1969.

〔9〕 B. Andrásfai, Introductory Graph theory, McGraw-Hill Book Company, New York, 1977.

〔10〕 Liu, C. L., Elements of Discrete Mathematics, chapter 4, McGraw-Hill, 1977.

〔11〕 卢开澄，图论及其应用，清华大学出版社，1981。

〔12〕 王朝瑞，图论，人民教育出版社，1981。

〔13〕 曹新谱，算法设计与分析，湖南科技出版社，1984。

〔14〕 严蔚敏、沈佩娟，数据结构，国防工业出版社，1981。

〔15〕 周以铨、肖位枢、刘玉，离散数学讲义，航空工业出版社，1987。